# U.S. Government

# Router

# Protection Profile

# For

# Medium Robustness Environments

**Information**
**Assurance**
**Directorate**

**Version 1.0**
**February 22, 2005**

**Protection Profile Title:**

1 U.S. Government Router Protection Profile for Medium Robustness Environments.

**Criteria Version:**

2 This Protection Profile (PP) was developed using Version 2.2 of the Common Criteria (CC) and applying the National Information Assurance Partnership (NIAP) interpretations that have been approved by Trust Technology Assessment Program/Common Criteria Evaluation Standard Scheme (TTAP/CCEVS) Management as of August 20, 2004.

# Table of Contents

# List of Tables and Figures

# 1 INTRODUCTION TO THE PROTECTION PROFILE

## 1.1 PP Identification

3 Title: U.S. Government Router PP for Medium Robustness Environments

4 Sponsor: National Security Agency (NSA)

5 CC Version: Common Criteria (CC) Version 2.2, and applicable interpretations.

6 Registration: <to be provided upon registration>

7 PP Version: Version 0.8 dated December 9, 2004.

8 Keywords: Router, protection profile, encryption, decryption, IPSEC ESP, IPSEC AH, IKE

## 1.2 Overview of the Protection Profile

9 The U.S. Government Router PP for Medium Robustness Environments specifies a set of security functional assurance requirements for Information Technology (IT) products. A router monitors, routes and manipulates network traffic to facilitate it's delivery for the proper destination on a network or between networks.

10 The Router PP is applicable to products regardless of whether they are externally or internally facing a given network. In addition, it addresses only security requirements and not any special considerations of any particular product design.

11 The Router PP was constructed to provide a target metric for the deployment of router devices. This protection profile identifies security functions and assurances that represent the lowest common set of requirements that must be addressed at a Medium Robustness level by a router.

12 The assurance requirements were originally based upon Evaluate Assurance Level (EAL) 4. In order to gain the necessary level of assurance for medium robustness environments, explicit requirements have been created for some families in the ADV class both to remove ambiguity in the existing ADV requirements and to provide greater assurance than that associated with EAL4. The assurance requirements are presented in Section 5.3.

13 This PP defines:

- assumptions about the security aspects of the environment in which the TOE will be used;
- threats that are to be addressed by the TOE;
- security objectives of the TOE and its environment;
- functional and assurances requirements to meet those security objectives; and;
- rationale demonstrating how the requirements meet the security objectives, and how the security objectives address the threats.

## 1.3 Conventions

14 Except for replacing United Kingdom spelling with American spelling, the notation, formatting, and conventions used in this PP are consistent with version 2.2 of the CC. Selected presentation choices are discussed here to aid the PP reader.

15 The CC allows several operations to be performed on functional requirements; *refinement*, *selection*, *assignment*, and *iteration* are defined in paragraph 2.1.4 of Part 2 of the CC. Each of these operations is used in this PP.

16 The **refinement** operation is used to add detail to a requirement, and thus further restricts a requirement. Refinement of security requirements is denoted by **bold text**.

17 The **selection** operation is used to select one or more options provided by the CC in stating a requirement. Selections that have been made by the PP authors are denoted by *italicized text*, selections to be filled in by the Security Target (ST) author appear in square brackets with an indication that a selection is to be made, [selection:], and are not italicized.

18 The **assignment** operation is used to assign a specific value to an unspecified parameter, such as the length of a password. Assignments that have been made by the PP authors are denoted by showing the value in square brackets, [Assignment_value], assignments to be filled in by the ST author appear in square brackets with an indication that an assignment is to be made [assignment:].

19 The **iteration** operation is used when a component is repeated with varying operations. Iteration is denoted by showing the iteration number in parenthesis following the component identifier, (iteration_number).

20 As this PP was sponsored, in part by NSA, NIAP, interpretations are used and are presented with the NIAP interpretation number as part of the requirement identifier (e.g., **FAU_GEN.1-NIAP-0407** for Audit data generation).

21 The CC paradigm also allows protection profile and security target authors to create their own requirements. Such requirements are termed "explicit requirements" and are permitted if the CC does not offer suitable requirements to meet the authors' needs. **Explicit requirements** must be identified and are required to use the CC class/family/component model in articulating the requirements. In this PP, explicit requirements will be indicated with the "(EXP)" following the component name.

22 Application Notes are provided to help the developer, either to clarify the intent of a requirement, identify implementation choices, or to define "pass-fail" criteria for a requirement. For those components where Application Notes are appropriate, the Application Notes will follow the requirement component.

## 1.4 Glossary of Terms

23 See Appendix B for the Glossary.

## 1.5 Document Organization

24 Section 1, Introduction to the Protection Profile, provides the document management and overview information necessary to identify the PP.

25 Section 38, Target of Evaluation (TOE) Description, defines the TOE and establishes the context of the TOE by referencing generalized security functions.

26 Section 3, Security Environment, describes the expected environment in which the TOE is to be used. This section defines the set of threats that are relevant to the secure operation of the TOE, organizational security policies with which the TOE must comply, and secure usage assumptions applicable to this analysis.

27 Section 4, Security Objectives, defines the set of security objectives to be satisfied by the TOE and by the TOE operating environment.

28 Section 5, IT Security Requirements, specifies the security functional and assurance requirements that must be satisfied by the TOE and the IT environment.

29 Section 6, Rationale, provides rationale to demonstrate that the security objectives satisfy the threats and policies. This section also explains how the set of requirements are complete relative to the security objectives and presents a set of arguments that address dependency analysis and Strength of Function (SOF) and use of the explicit requirement.

30 Section 7, Appendices, includes the appendices that accompany the PP and provides clarity and/or explanation for the reader.

31 Appendix A, References, provides background material for further investigation by users of the PP.

32 Appendix B, Glossary, provides a listing of definitions of terms.

33 Appendix C, Acronyms, provides a listing of acronyms used throughout the document.

34 Appendix D, Robustness Environment Characterization, contains a discussion characterizing the level of robustness TOEs compliant with the PP can achieve. The PPRB created a discussion that provides a definition of factors for TOE environments and an explanation of how a given level of robustness is categorized.

35 Appendix E, Explanatory Material for Explicit Assurance Requirements, provides objectives and application notes for the explicit ADV requirements contained in this PP.

36 Appendix F, Refinements, identifies the refinements that were made to CC requirements where text is deleted from a requirement.

37 Appendix G, Statistical Number Generator Tests, describes the statistical tests that must be performed to the random number generators.

38 Appendix H, Randomizer Qualification Testing Requirements, lists the randomiser qualification statistical test suite and describes the randomiser qualification test process.

# 2 TOE DESCRIPTION

39 This Protection Profile specifies the minimum security requirements to satisfy Medium Robustness Environments for a TOE that is a router.

## 2.1 Product Type

40 Router PP-conformant products support the ability to monitor, route, and manipulate network traffic to facilitate its delivery to the proper destination on a network or between networks.

41 The Router PP was constructed to provide a target and metric for the deployment of router devices. This protection profile identifies security functions and assurances that represent the minimum set of security requirements that should be addressed at a Medium Robustness level by a router.

42 The Router PP is applicable to products regardless of whether they are externally or internally facing a given network. In addition, it addresses only security requirements and not any special considerations of any particular product design.

43 The Router PP addresses only those factors that should be considered when dealing with a dedicated router. It does not cover extra functionality that may be added to a router (such as point-to-point user network data encryption and detailed traffic monitoring and manipulation) that in essence changes the router to another type of device such as a Virtual Private Network (VPN) endpoint or a firewall. Those devices should be covered under their own appropriate protection profile document.

## 2.2 TOE Definition

44 A router is a network-layer device (layer 3 under the Open Systems Interconnect (OSI) model) that connects networks that use the same network-layer protocol, for example, Transmission Control Protocol/Internet Protocol (TCP/IP) or Internetwork Packet Exchange (IPX). A router uses standardized protocols, such as Routing Information Protocol (RIP), Border Gateway Protocol (BGP), or Intermediate System to Intermediate System (IS-IS) to move packets efficiently to their destination. A router can be configured to support multiple network protocols and routing protocols on a single device.

## 2.3 General TOE Functionality

45 Internet Protocol (IP) routing is a connectionless means of transferring information contained within variable length packets. The delivery of traffic between the host and destination is generally best effort traffic and delivery is not guaranteed. Since prior logical paths are not established, each IP packet can be dynamically routed across multiple paths. A router dynamically determines the best path based on assigned routing protocols and the status of the network.

46 International Organization for Standardization (ISO) Connectionless Network Protocol (CLNP) routing is an OSI network layer protocol that carries upper-layer data and error indications over connectionless links. CLNP provides the interface between the Connectionless Network Service (CLNS) and upper layers. The IS-IS routing protocol IS-IS is used to dynamically determines the best path to route traffic over a CLNP network.

47 Internetwork Packet eXchange (IPX) is also a connectionless means of transferring information contained within variable length packets. The delivery of traffic between the host and destination is generally best effort traffic and delivery is not guaranteed. Since prior logical paths are not established, each IPX packet can be dynamically routed across multiple paths. A router dynamically determines the best path based on assigned routing protocols and the status of the network.

48 Trusted paths must be established between the router and the management station and (trusted) channels must be established between individual routers in order to exchange management information. Between routers, network control information is exchanged via (trusted) channels to allow dynamic connection establishment and packet routing. Network control information consists of specific requests and instructions that include destination address, routing controls, and signalling information. Examples of control information in the IP environment include Open Shortest Path First (OSPF), BGP, Resource Reservation Protocol (RSVP), and Label Distribution Protocol (LDP).

49 A router that is compliant with the Router PP provides the following security functions in its evaluated configuration:

- Audit – Section 5.1.1 “Security Audit (FAU)” describes the TOE’s generation of auditable events, audit records, alarms and audit management. Table 7 in the FAU_GEN.1-NIAP-0410 requirement lists the minimum set of auditable events that must be available to the Security Administrator for configuration on the TOE. Each auditable event must generate an audit record. Table 7 also provides a minimum list of attributes that must be included in each audit record. The ST author may include additional auditable events and audit record attributes. If the ST author includes any additional functional requirements not specified by this PP, they must consider any security relevant events associated with those requirements and include them in the TOE’s list of auditable events and records. In addition to generating auditable events, the TOE must monitor their occurrences and provide a Security Administrator configurable threshold for determining a potential security violation. Once the TOE has detected a potential security violation, an alarm is generated and a message is displayed at the TOE’s

local console and each active remote administrator console (all administrative roles included). Additionally, the Security Administrator can configure the TOE to generate an audible alarm to indicate a potential security violation. If an administrator console is not active, the TOE stores the message for display when the console becomes active (e.g., when the administrator establishes a remote session to the TOE). The message must contain the potential security violation and all audit records associated with the potential security violation. The message will be displayed at the various consoles until administrator acknowledgement of the message has occurred. As mentioned in the "Administrative" section above, the Audit Administrator's role is restricted to viewing the contents of the audit records and the deletion of the audit trail. The TOE does provide the Audit Administrator with a sorting and searching capability to improve audit analysis. The Security Administrator configures auditable events, backs-up and deletes audit data, and manages audit data storage. The TOE provides the Security Administrator with a configurable audit trail threshold to track the storage capacity of the audit trail. Once the threshold is met, the TOE generates an alarm and displays a message in the same fashion as described above, including the option of the audible alarm. In addition to displaying the message, the Security Administrator may configure the TOE to prevent all auditable events except for those performed by the Security and Audit Administrators or overwrite the oldest audit records in the audit trail.

- Encryption – Cryptographic algorithms and key management functions that meet published standards are required in Router PP-complaint products. Section 5.1.2 "Cryptographic Support" defines the minimum set of cryptographic attributes required by the TOE. The TOE's cryptographic module(s) must be FIPS PUB 140-2 validated and must meet, as a minimum, the security requirements of "Security Level 1". The ST author may implement the cryptographic module(s) in hardware, software, or a combination of both. The TOE must generate and distribute symmetric and asymmetric keys. The ST author is provided several implementation selections for key generation and may distribute keys manually, electronically, or both. The TOE must perform data encryption/decryption using the Triple Data Encryption Algorithm (TDEA) with a minimum key size of 168 bits or the Advanced Encryption Standard (AES) with a minimum key size of 192 bits. Additional requirements for key destruction, digital signature generation/verification, random number generation and cryptographic hashing and message authentication are provided in section 5.1.2.

- Identification and Authentication – The TOE requires multiple Identification and Authentication (I&A) mechanisms for access to services residing on the TOE or for services mediated by the TOE. The type of authentication mechanism required depends on the origin of the source (i.e., remote user, TOE console) requesting the service.
- Administration-"Administrators" refers to the roles assigned to the individuals responsible for the installation, configuration, and maintenance of the TOE. The TOE requires three separate administrative roles: Cryptographic Administrator, Audit Administrator and Security Administrator. The Cryptographic Administrator is responsible for the configuration and maintenance of cryptographic elements related to the establishment of secure connections to and from the TOE. The Audit Administrator is responsible for the regular review of the TOE's audit data. The Security Administrator is responsible for all other administrative tasks (e.g., creating the TOE security policy) not addressed by the other two administrative roles. It is important to note that while this PP requires the three administrative roles outlined above, it provides the ST author the option of including additional administrative roles as well.
- Trusted Channel/Trusted Path- The TOE is required to provide two types of encrypted communications: trusted channel and trusted path. Trusted channel refers to the encrypted connection between the TOE and a trusted IT entity. Trusted path refers to the encrypted connection used to authenticate an administrator with the TOE.

## 2.4 TOE Operation Environment

50 A router is placed at the edge of a given network or network segment. For a router to function it must have at least two distinct networks or network segments to pass data between.

# 3 SECURITY ENVIRONMENT

51 A medium robustness TOE is considered sufficient protection for environments where the likelihood of an attempted compromise is medium. This implies that the motivation of the threat agents will be average in environments that are suitable for TOEs of medium robustness. Note that this also implies that the resources and expertise of the threat agents really are not factors that need to be considered, because highly sophisticated threat agents will not be motivated to use great expertise or extensive resources in an environment where medium robustness is suitable.

52 The medium motivation of the threat agents can be reflected in a variety of ways. One possibility is that the value of the data processed or protected by the TOE will be only medium, thus providing little motivation of even a totally unauthorized entity to attempt to compromise the data. Another possibility, (where higher value data is processed or protected by the TOE) is that the procuring organization will provide environmental controls (that is, controls that the TOE itself does not enforce) in order to ensure that threat agents that have generally high motivation levels (because of the value of the data) cannot logically or physically access the TOE (e.g., all users are "vetted" to help ensure their trustworthiness, and connectivity to the TOE is restricted).

53 The remainder of this section addresses the following:

- Threats to TOE assets or to the TOE environment which must be countered;
- Organizational Security Policies that compliant TOEs must enforce; and
- Assumptions about the security aspects of a compliant TOE environment.

54 It is important to note to vendors and end users that any IT entity that is used to protect National Security information, and employs cryptography as a protection mechanism, will require the TOE's key management techniques to be approved by NSA prior to the fielding of the TOE.

## 3.1 Threats

### 3.1.1 Threat Agent Characterization

55 In addition to helping define the robustness appropriate for a given environment, the threat agent is a key component of the formal threat statements in the PP. Threat agents are typically characterized by a number of factors such as *expertise*, *available resources*, and *motivation*. Because each robustness level is associated with a variety of environments, there are corresponding varieties of specific threat agents (that is, the threat agents will have different combinations of motivation, expertise, and available resources) that are valid for a given level of robustness.

The following discussion explores the impact of each of the threat agent factors on the ability of the TOE to protect itself (that is, the robustness required of the TOE).

56 The *motivation* of the threat agent seems to be the primary factor of the three characteristics of threat agents outlined above. Given the same expertise and set of resources, an attacker with low motivation may not be as likely to attempt to compromise the TOE. For example, an entity with no authorization to low value data none-the-less has low motivation to compromise the data; thus a basic robustness TOE should offer sufficient protection. Likewise, the fully authorized user with access to highly valued data similarly has low motivation to attempt to compromise the data, thus again a basic robustness TOE should be sufficient.

57 Unlike the motivation factor, however, the same cannot be said for *expertise*. A threat agent with low motivation and low expertise is just as unlikely to attempt to compromise a TOE as an attacker with low motivation and high expertise; this is because the attacker with high expertise does not have the motivation to compromise the TOE even though they may have the expertise to do so. The same argument can be made for *resources* as well.

58 Therefore, when assessing the robustness needed for a TOE, the motivation of threat agents should be considered a "high water mark". That is, *the robustness of the TOE should increase as the motivation of the threat agents increases.*

59 Having said that, the relationship between expertise and resources is somewhat more complicated. In general, if resources include factors other than just raw processing power (money, for example), then expertise should be considered to be at the same "level" (low, medium, high, for example) as the resources because money can be used to purchase expertise. Expertise in some ways is different, because expertise in and of itself does not automatically procure resources. However, it may be plausible that someone with high expertise can procure the requisite amount of resources by virtue of that expertise (for example, hacking into a bank to obtain money in order to obtain other resources).

60 It may not make sense to distinguish between these two factors; in general, it appears that the only effect these may have is to lower the robustness requirements. For instance, suppose an organization determines that, because of the value of the resources processed by the TOE and the trustworthiness of the entities that can access the TOE, the motivation of those entities would be "medium". This normally indicates that a medium robustness TOE would be required because the likelihood that those entities would attempt to compromise the TOE to get at those resources is in the "medium" range. However, now suppose the organization determines that the entities (threat agents) that are the least trustworthy have no resources and are unsophisticated. In this case, even though those threat agents have medium motivation, the likelihood that they would be able to mount a

successful attack on the TOE would be low, and so a basic robustness TOE may be sufficient to counter that threat.

61 It should be clear from this discussion that there is no "cookbook" or mathematical answer to the question of how to specify exactly the level of motivation, the amount of resources, and the degree of expertise for a threat agent so that the robustness level of TOEs facing those threat agents can be rigorously determined. However, an organization can look at combinations of these factors and obtain a good understanding of the likelihood of a successful attack being attempted against the TOE. Each organization wishing to procure a TOE must look at the threat factors applicable to their environment; discuss the issues raised in the previous paragraph; consult with appropriate accreditation authorities for input; and document their decision regarding likely threat agents in their environment.

62 The important general points are:

- The motivation for the threat agent defines the upper bound with respect to the level of robustness required for the TOE.
- A threat agent's expertise and/or resources that are "lower" than the threat agent's motivation (e.g., a threat agent with high motivation but little expertise and few resources) may lessen the robustness requirements for the TOE (see next point, however).
- The availability of attacks associated with high expertise and/or high availability of resources (for example, via the Internet or "hacker chat rooms") introduces a problem when trying to define the expertise of, or resources available to, a threat agent.

63 The following threats are addressed by the TOE and should be read in conjunction with the threat rationale, Section 6.1. There are other threats that the TOE does not address (e.g., malicious developer inserting a backdoor into the TOE) and it is up to a site to determine how these types of threats apply to its environment.

**Table 1 Medium Robustness Applicable Threats**

| Threat Name | Threat Definition |
|---|---|
| T.ADMIN_ERROR | An administrator may incorrectly install or configure the TOE, or install a corrupted TOE resulting in ineffective security mechanisms. |
| T.ADMIN_ROGUE | An administrator's intentions may become malicious resulting in user or TOE Security Functions (TSF) data being compromised. |
| T.AUDIT_COMPROMISE | A malicious user or process may view audit records, cause audit records to be lost or modified, or prevent future audit records from being recorded, thus masking a user's action. |
| T.CRYPTO_COMPROMISE | A malicious user or process may cause key, data or executable code associated with the cryptographic functionality to be inappropriately accessed (viewed, modified, or deleted), thus compromising the cryptographic mechanisms and the data protected by those mechanisms. |
| T.FLAWED_DESIGN | Unintentional or intentional errors in requirements specification or design of the TOE may occur, leading to flaws that may be exploited by a malicious user or program. |
| T.FLAWED_IMPLEMENTATION | Unintentional or intentional errors in implementation of the TOE design may occur, leading to flaws that may be exploited by a malicious user or program. |
| T.MALICIOUS_TSF_COMPROMISE | A malicious user or process may cause TSF data or executable code to be inappropriately accessed (viewed, modified, or deleted). |

| | |
|---|---|
| T.MASQUERADE | A malicious user, process, or external IT entity may masquerade as an authorized entity in order to gain access to data or TOE resources. |
| T.POOR_TEST | Lack of or insufficient tests to demonstrate that all TOE security functions operate correctly (including in a fielded TOE) may result in incorrect TOE behavior being undiscovered thereby causing potential security vulnerabilities. |
| T.REPLAY | A user may gain inappropriate access to the TOE by replaying authentication information, or may cause the TOE to be inappropriately configured by replaying TSF data or security attributes (e.g., captured as transmitted during the course of legitimate use). |
| T.RESIDUAL_DATA | A user or process may gain unauthorized access to data through reallocation of TOE resources from one user or process to another. |
| T.RESOURCE_EXHAUSTION | A malicious process or user may block others from system resources (e.g., *connection state tables, TCP connections) via* a resource exhaustion denial of service attack. |
| T.SPOOFING | A malicious user, process, or external IT entity may misrepresent itself as the TOE to obtain identification and authentication data. |
| T.UNATTENDED_SESSION | A user may gain unauthorized access to an unattended session. |
| T.UNAUTHORIZED_ACCESS | A user may gain access to user data for which they are not authorized according to the TOE security policy. |

| | |
|---|---|
| T.UNIDENTIFIED_ACTIONS | The administrator may fail to notice potential security violations, thus limiting the administrator's ability to identify and take action against a possible security breach. |
| T.UNAUTHORIZED_PEER | An unauthorized IT entity may attempt to establish a security association with the TOE. |
| T.UNKNOWN_STATE | When the TOE is initially started or restarted after a failure, the security state of the TOE may be unknown. |
| T.TRAFFIC_ANALYSIS | An attacker collects source and destination addresses, volume of data, and time of day that messages are sent. |

## 3.2 Organizational Security Policies

64 An organizational security policy is a set of rules, practices, and procedures imposed by an organization to address its security needs.

**Table 2 Medium Robustness Applicable Policies**

| Policy Name | Policy Definition |
|---|---|
| P.ACCESS_BANNER | The TOE shall display an initial banner describing restrictions of use, legal agreements, or any other appropriate information to which users consent by accessing the TOE. |
| P.ACCOUNTABILITY | The authorized users of the TOE shall be held accountable for their actions within the TOE. |
| P.ADMIN_ACCESS | Administrators shall be able to administer the TOE both locally and remotely through protected communications channels. |
| P.CRYPTOGRAPHIC_FUNCTIONS | The TOE shall provide cryptographic functions for its own use, including encryption/decryption and digital signature operations. |
| P.CRYPTOGRAPHY_VALIDATED | Where the TOE requires FIPS-approved security functions, only National Institute of Standards Technology Federal Information Processing Standard Publication (NIST FIPS) validated cryptography (methods and implementations) are acceptable for key management (i.e., generation, access, distribution, destruction, handling, and storage of keys) and cryptographic services (i.e., encryption, decryption, signature, hashing, key distribution, and random number generation services). |
| P.VULNERABILITY_ANALYSIS_TEST | The TOE must undergo appropriate independent vulnerability analysis and penetration testing to demonstrate that the TOE is resistant to an attacker possessing a medium attack potential. |

| | |
|---|---|
| P.COMPATIBILITY | The TOE must meet Request for Comments (RFC) requirements for implemented protocols to facilitate inter-operation with other routers and network equipment using the same protocols. |

## 3.3 Assumptions

65 This section contains assumptions regarding the security environment and the intended usage of the TOE.

**Table 3 Medium Robustness Applicable Assumptions**

| Assumption Name | Assumption Definition |
|---|---|
| A.NO_GENERAL_PURPOSE | The administrator ensures there are no general-purpose computing or storage repository capabilities (e.g., compilers, editors, or user applications) available on the TOE. |
| A.PHYSICAL | It is assumed that the IT environment provides the TOE with appropriate physical security, commensurate with the value of the IT assets protected by the TOE. |
| A.AVAILABILITY | Network resources shall be available to allow clients to satisfy mission requirements and to transmit information. |

# 4 SECURITY OBJECTIVES

66 This section identifies the security objectives of the TOE and its supporting environment. The security objectives identify the responsibilities of the TOE and its environment in meeting the security needs.

## 4.1 TOE Security Objectives

**Table 4 Medium Robustness Security Objectives**

| Objective Name | Objective Definition |
|---|---|
| O.ADMIN_ROLE | The TOE will provide administrator roles to isolate administrative actions, and to make the administrative functions available locally and remotely. |
| O.AUDIT_GENERATION | The TOE will provide the capability to detect and create records of security-relevant events associated with users. |
| O.AUDIT_PROTECTION | The TOE will provide the capability to protect audit information. |
| O.AUDIT_REVIEW | The TOE will provide the capability to selectively view audit information, and alert the administrator of identified potential security violations. |
| O.CHANGE_MANAGEMENT | The configuration of, and all changes to, the TOE and its development evidence will be analyzed, tracked, and controlled throughout the TOE's development. |
| O.CORRECT_ TSF_OPERATION | The TOE will provide a capability to test the TSF to ensure the correct operation of the TSF in its operational environment. |
| O.DISPLAY_BANNER | The TOE will display an advisory warning regarding use of the TOE. |
| O.DOCUMENT_KEY_LEAKAGE | The bandwidth of channels that can be used to compromise key material shall be documented. |
| O.MAINT_MODE | The TOE shall provide a mode from which recovery or initial startup procedures can be performed. |

| | |
|---|---|
| O.MANAGE | The TOE will provide all the functions and facilities necessary to support the administrators in their management of the security of the TOE, and restrict these functions and facilities from unauthorized use. |
| O.MEDIATE_INFORMATION_FLOW | The TOE must mediate the flow of information between sets of TOE network interfaces or between a network interface and the TOE itself in accordance with its security policy. |
| O.PEER_AUTHENTICATION | The TOE will authenticate each peer TOE that attempts to establish a security association with the TOE. |
| O.PROTOCOLS | The TOE will ensure that standardized protocols are implemented in the TOE to RFC and/or Industry specifications to ensure interoperability. |
| O.REPLAY_DETECTION | The TOE will provide a means to detect and reject the replay of authentication data and other TSF data and security attributes. |
| O.RESIDUAL_INFORMATION | The TOE will ensure that any information contained in a protected resource is not released when the resource is reallocated. |
| O.RESOURCE_SHARING | The TOE shall provide mechanisms that mitigate attempts to exhaust connection-oriented resources provided by the TOE (e.g., entries in a connection state table; TCP connections to the TOE). |
| O. ROBUST_ADMIN_GUIDANCE | The TOE will provide administrators with the necessary information for secure delivery and management. |

| | |
|---|---|
| O.ROBUST_TOE_ACCESS | The TOE will provide mechanisms that control a user's logical access to the TOE and to explicitly deny access to specific users when appropriate. |
| O.SELF_PROTECTION | The TSF will maintain a domain for its own execution that protects itself and its resources from external interference, tampering or unauthorized disclosure. |
| O.SOUND_DESIGN | The TOE will be designed using sound design principles and techniques. The TOE design, design principles and design techniques will be adequately and accurately documented. |
| O.SOUND_IMPLEMENTATION | The implementation of the TOE will be an accurate instantiation of its design, and is adequately and accurately documented. |
| O.THOROUGH_FUNCTIONAL_TESTING | The TOE will undergo appropriate security functional testing that demonstrates the TSF satisfies the security functional requirements. |
| O.TIME_STAMPS | The TOE shall provide reliable time stamps and the capability for the administrator to set the time used for these time stamps. |
| O.TRUSTED_PATH | The TOE will provide a means to ensure that users are not communicating with some other entity pretending to be the TOE when supplying identification and authentication data. |
| O.USER_GUIDANCE | The TOE will provide users with the information necessary to correctly use the security mechanisms. |

| | |
|---|---|
| O.VULNERABILITY_ANALYSIS_TEST | The TOE will undergo appropriate independent vulnerability analysis and penetration testing to demonstrate the design and implementation of the TOE and does not allow attackers with medium attack potential to violate the TOE's security policies. |

## 4.2 Environment Security Objectives

**Table 5 Medium Robustness Environmental Security Objectives**

| Environmental Objective Name | Environmental Objective Definition |
|---|---|
| OE.NO_GENERAL_PURPOSE | The Administrator ensures there are no general-purpose computing or storage repository capabilities (e.g., compilers, editors, or user applications) available on the TOE. |
| OE.PHYSICAL | Physical security, commensurate with the value of the TOE and the data it contains, is assumed to be provided by the IT environment. |
| OE.AVAILABILITY | Network resources will be available to allow clients to satisfy mission requirements and to transmit information. |

# 5 IT SECURITY REQUIREMENTS

## 5.1 TOE Security Functional Requirements

67 This section defines the functional requirements for the TOE. Functional requirements in this PP were drawn directly from Part 2 of the CC, or were based on Part 2 of the CC. These requirements are relevant to supporting the secure operation of the TOE.

**Table 6 Security Functional Requirements**

| Functional Components (from CC Part 2) | |
|---|---|
| FAU_ARP.1 | Security alarms |
| FAU_ARP_ACK_(EXP).1 | Security alarm acknowledgement |
| FAU_GEN.1-NIAP-0407 | Audit data generation |
| FAU-GEN.2-NIAP-0410 | User identity association |
| FAU_SAA.1-NIAP-0407 | Potential violation analysis |
| FAU_SAR.1 | Audit review |
| FAU_SAR.2 | Restricted audit review |
| FAU_SAR.3 | Selectable audit review |
| FAU_SEL.1-NIAP-0407 | Audit event selection |
| FAU_STG.NIAP-0414-1-NIAP-0429 | Site-configurable prevention of audit loss |
| FAU_STG.1-NIAP-0429 | Audit event storage |
| FAU_STG.3 | Action in case of possible audit data loss |
| FCS_BCM_(EXP).1 | Baseline Cryptographic Module |
| FCS_CKM.1(1) | Cryptographic key generation (for symmetric keys using Random Number Generator(RNG)) |

| | |
|---|---|
| FCS_CKM.1(2) | Cryptographic key generation (for asymmetric keys) |
| FCS_CKM.2 | Cryptographic key distribution |
| FCS_CKM.4 | Cryptographic key destruction |
| FCS_CKM_(EXP).1 | Cryptographic Key Validation and Packaging |
| FCS_CKM_(EXP).2 | Cryptographic Key Validation and Storage |
| FCS_COA_(EXP).1 | Cryptographic Operations Availability |
| FCS_COP.1(1) | Cryptographic operation (for data encryption/decryption) |
| FCS_COP.1(2) | Cryptographic operation (for cryptographic signature) |
| FCS_COP.1(3) | Cryptographic operation (for cryptographic hashing) |
| FCS_COP.1(4) | Cryptographic operation (for message authentication) |
| FCS_COP.1(5) | Cryptographic operation (for cryptographic key agreement) |
| FCS_COP_(EXP).1 | Random Number Generation |
| FCS_IKE_(EXP).1 | Internet Key Exchange |
| FDP_IFC.1(1) | Subset information flow control (unauthenticated TOE services policy) |
| FDP_IFC.1(2) | Subset information flow control (authenticated TOE services policy) |
| FDP_IFF.1(1) | Simple security attributes (unauthenticated policy) |
| FDP_IFF.1(2) | Simple security attributes (authenticated policy) |
| FDP_RIP.2 | Full residual information protection |
| FIA_AFL.1 | Authentication failures |
| FIA_ATD.1(1) | User attribute definition (Human users) |

| | |
|---|---|
| FIA_ATD.1(2) | User attribute definition (TOE to TOE Identification) |
| FIA_UAU.2 | User authentication before any action |
| FIA_UAU_(EXP).5 | Authentication Mechanism |
| FIA_UID.2 | User identification before any action |
| FIA_USB.1 | User-Subject Binding |
| FMT_MOF.1(1) | Management of security functions behavior (TSF non-cryptographic self-test) |
| FMT_MOF.1(2) | Management of security functions behavior (cryptographic self-test) |
| FMT_MOF.1(3) | Management of security functions behavior (Audit Review) |
| FMT_MOF.1(4) | Management of security functions behavior (Audit Selection) |
| FMT_MOF.1(5) | Management of security functions behavior (Alarms) |
| FMT_MOF.1(6) | Management of security functions behavior (quota mechanism) |
| FMT_MSA.1(1) | Management of security attributes (unauthenticated) |
| FMT_MSA.1(2) | Management of security attributes (authenticated) |
| FMT_MSA.3(1) | Static attribute initialization (unauthenticated services) |
| FMT_MSA.3(2) | Static attribute initialization (authenticate services) |
| FMT_MTD.1(1) | Management of TSF data (non-cryptographic, non-time TSF data) |
| FMT_MTD.1(2) | Management of TSF data (cryptographic TSF data) |
| FMT_MTD.1(3) | Management of TSF data (time TSF data) |
| FMT_MTD.1(4) | Management of TSF data (Router Policy Ruleset) |

| | |
|---|---|
| FMT_MTD.2(1) | Management of limits on TSF data (transport-layer quotas) |
| FMT_MTD.2(2) | Management of limits on TSF data (controlled connection-oriented quotas) |
| FMT_REV.1 | Revocation |
| FMT_SMF.1 | Specification of Management Functions |
| FMT_SMR.2 | Restrictions on security roles |
| FPT_FLS.1 | Failure with preservation of secure state |
| FPT_ITA.1 | Inter-TSF availability within a defined availability metric |
| FPT_ITC.1 | Inter-TSF confidentiality during transmission |
| FPT_ITI.1 | Inter-TSF detection of modification |
| FPT_PRO_(EXP).1 | Standard protocol usage |
| FPT_RCV.2 | Automated Recovery |
| FPT_RPL.1 | Replay detection |
| FPT_RVM.1 | Non-bypassability of the TSP |
| FPT_SEP.2 | Security Function Policy (SFP) domain separation |
| FPT_STM.1 | Reliable time stamps |
| FPT_TDC.1 | Inter-TSF basic TSF data consistency |
| FPT_TST_(EXP).4 | TSF testing (with cryptographic integrity verification) |
| FPT_TST_(EXP).5 | Cryptographic self-test |
| FRU_RSA.1(1) | Maximum quotas (transport-layer quotas) |
| FRU_RSA.1(2) | Maximum quotas (controlled connection-oriented quotas) |

| | |
|---|---|
| FTA_SSL.3 | TSF-initiated termination |
| FTA_TAB.1 | Default TOE access banners |
| FTA_TSE.1 | TOE session establishment |
| FTP_ITC.1(1) | Inter-TSF trusted channel (Prevention of Disclosure) |
| FTP_ITC.1(2) | Inter-TSF trusted channel (Detection of Modification) |
| FTP_TRP.1(1) | Trusted path (Prevention of Disclosure) |
| FTP_TRP.1(2) | Trusted path (Detection of Modification) |

### 5.1.1 Security Audit (FAU)

#### 5.1.1.1 Security alarms (FAU_ARP.1)

FAU_ARP.1.1 The TSF shall [immediately display a message identifying the potential security violation, and make accessible the audit record contents associated with the auditable event(s) that generated the alarm, at the

a) local console;

b) remote Security Administrator sessions that exist;

c) remote Security Administrator sessions that are initiated before the alarm has been acknowledged; and

d) [selection: [ST assignment: other methods determined by the ST author], no other methods]

] upon detection of a potential security violation.

68 *Application Note: The TSF provides a message to the local console regardless of whether an administrator is logged in. The message is displayed at the remote console if an administrator is already logged in, or when an administrator logs in if the alarm message has not been acknowledged. The audit records contents associated with the alarm may or may not be part of the message displayed; however the relevant audit information must be available to administrators. In addition, the TOE provides an audible alarm that can be configured to sound an*

*alarm if desired by the Security Administrator. It is acceptable for the ST author to fill the open assignment without any, if no other methods (e.g., pager, e-mail) are included in the TOE.*

### 5.1.1.2 Security alarm acknowledgement (FAU_ARP_ACK_ (EXP).1)

FAU_ARP_ACK_(EXP).1.1 – The TSF shall display the alarm message identifying the potential security violation and make accessible the audit record contents associated with the auditable event(s) until it has been acknowledged. An audible alarm will sound until acknowledged by an administrator.

FAU_ARP_ACK_(EXP).1.2 – The TSF shall display an acknowledgement message identifying a reference to the potential security violation, a notice that it has been acknowledged, the time of the acknowledgement and the user identifier that acknowledged the alarm, at the:

- local console, and
- remote administrator sessions that received the alarm.

69 *Application Note: This explicit requirement is necessary since a CC requirement does not exist to ensure an administrator will be aware of the alarm. The intent is to ensure that if an administrator is logged in and not physically at the console or remote workstation the message will remain displayed until they have acknowledged it. The message will not be scrolled off the screen due to other activity-taking place (e.g., the Audit Administrator is running an audit report). If the Security Administrator configures the TOE to generate an audible alarm, the alarm will sound until an administrator acknowledges the alarm. Acknowledging the message and audible alarm could be a single event, or different events.*

70 *FAU_ARP_ACK_(EXP).1.2 ensures that each administrator that received the alarm message also receives the acknowledgement message, which includes some form of reference to the alarm message, who acknowledged the message and when.*

### 5.1.1.3 Audit data generation (FAU_GEN.1-NIAP-0407)

FAU_GEN.1.1-NIAP-0407 – The TSF shall be able to generate an audit record of the following auditable events:

a) Start-up and shutdown of the audit functions;

b) All auditable events listed in Table 10;

c) [selection: [assignment: events at a basic level of audit introduced by the inclusion of additional SFRs determined by the ST Author], [assignment: events

commensurate with a basic level of audit introduced by the inclusion of explicit requirements determined by the ST Author], no additional events].

71 *Application Note: For the first assignment in the selection, the ST author augments the table (or lists explicitly) the audit events associated with the basic level of audit for any SFRs that the ST author includes that are not included in this PP.*

72 *Likewise, for the second assignment the ST author includes audit events that may arise due to the inclusion of any explicit requirements not already in the PP. Because "basic" audit is not defined for such requirements, the ST author will need to determine a set of events that are commensurate with the type of information that is captured at the basic level for similar requirements. It is acceptable for the ST author to choose "no additional events", if the ST author has not included additional requirements, or has included additional requirements that do not have a basic level (or commensurate level) of audit associated with them.*

FAU_GEN.1.2-NIAP-0407 - The TSF shall record within each audit record at least the following information:

a) Date and time of the event, type of event, and the outcome (success or failure) of the event; and

b) For each audit event type, based on the auditable event definitions of the functional components included in the PP/ST, [information specified in column three of Table 7 below].

73 *Application Note: In column 3 of the table below, "if applicable" is used to designate data that should be included in the audit record if it "makes sense" in the context of the event that generates the record. For example, in FDP_IFF, packets may be allowed to flow that do not have a transport layer component (e.g., an Internet Control Message Protocol (ICMP) Echo request). For those packets, there is nothing to record with respect to the transport layer abstractions.*

**Table 7 Auditable Events Table**

| Requirement | Auditable Events | Audit Record Contents |
|---|---|---|
| FAU_ARP.1 | Actions taken due to imminent security violations. | Identification of what caused the generation of the alarm. |

| | | |
|---|---|---|
| FAU_ARP_ACK_(EXP).1 | Actions taken due to imminent security violations. | The identity and location of the administrator that acknowledged the alarm. |
| FAU_GEN.1-NIAP-0407 | None. | |
| FAU_GEN.2-NIAP-0410 | None. | |
| FAU_SAA.1-NIAP-0407 | Enabling and disabling of any of the analysis mechanisms;<br>Automated responses performed by the tool. | The identity of the Security Administrator performing the function. |
| FAU_SAR.1 | Reading of information from the audit records. | The identity of the Audit Administrator performing the function. |
| FAU_SAR.2 | Unsuccessful attempts to read information from the audit records. | The identity of the administrator performing the function. |
| FAU_SAR.3 | None. | |
| FAU_SEL.1-NIAP-0407 | All modifications to the audit configuration that occur while the audit collection functions are operating. | The identity of the Security Administrator performing the function. |
| FAU_STG.NIAP-0414-1-NIAP-0429 | Actions taken due to the audit storage failure. | The identity of the Security Administrator performing the function. |
| FAU_STG.1-NIAP-0429 | None. | |

| | | |
|---|---|---|
| FAU_STG.3 | Actions taken due to exceeding the audit threshold. | The identity of the Security Administrator performing the function. |
| FCS_BCM_(EXP).1 | None. | |
| FCS_CKM.1(1) | a) Failure of the activity;<br>b) Generation and loading of key. | |
| FCS_CKM.1(2) | a) Failure of the activity;<br>b) Generation and loading of key pair for digital signatures. | |
| FCS_CKM.2 | a) Failure of the activity;<br>b) Generation and loading of key. | |
| FCS_CKM.4 | a) Failure of the activity;<br>b) Generation and loading of key. | |
| FCS_CKM.(EXP).1 | a) Failure of the activity;<br>b) Generation and loading of key. | |
| FCS_CKM.(EXP).2 | a) Failure of the activity;<br>b) Generation and loading of key. | |

| | | |
|---|---|---|
| FCS_COA_(EXP).1 | None. | |
| FCS_COP.1(1) | Failure of cryptographic operation. | Type of cryptographic operation.<br>Any applicable cryptographic mode(s) of operation, excluding any sensitive information. |
| FCS_COP.1(2) | Failure of cryptographic operation. | Type of cryptographic operation<br>Any applicable cryptographic mode(s) of operation, excluding any sensitive information. |
| FCS_COP.1(3) | Failure of cryptographic operation. | Type of cryptographic operation.<br>Any applicable cryptographic mode(s) of operation, excluding any sensitive information. |
| FCS_COP.1(4) | Failure of cryptographic operation. | Type of cryptographic operation.<br>Any applicable cryptographic mode(s) of operation, excluding any sensitive information. |
| FCS_COP.1(5) | Failure of cryptographic operation. | Type of cryptographic operation.<br>Any applicable cryptographic mode(s) of operation, excluding any sensitive information. |

| | | |
|---|---|---|
| FCS_COP_(EXP).1 | Failure of cryptographic operation. | Type of cryptographic operation.<br>Any applicable cryptographic mode(s) of operation, excluding any sensitive information. |
| FCS_IKE_(EXP).1 | a) Generation and loading of key pair for digital signatures;<br>b) Changes to the pre-shared key used for authentication;<br>c) All modifications to the key lifetimes;<br>d) Failure of the authentication in Phase 1;<br>e) Failure to negotiate a security association in Phase 2. | If failure occurs, record a descriptive reason for the failure. |
| FDP_IFC.1(1) | None. | |
| FDP_IFC.1(2) | None. | |

| | | |
|---|---|---|
| FDP_IFF.1(1) | a) Decisions to permit or deny information flows;<br>b) Operation applied to each information flow permitted. | Presumed identity of source subject.<br>Identity of destination subject.<br>Transport layer protocol, if applicable.<br>Source subject service identifier, if applicable.<br>Destination subject service identifier, if applicable.<br>Identity of the interface on which the TOE received the packet.<br>For denied information flows, the reason for denial. |
| FDP_IFF.1(2) | Decisions to permit or deny information flows. | Presumed identity of source subject.<br>Identity of destination subject.<br>Transport layer protocol, if applicable.<br>Source subject service identifier, if applicable.<br>Destination subject service identifier, if applicable.<br>Identity of the interface on which the TOE received the packet.<br>For denied information flows, the reason for denial. |
| FDP_RIP.2 | None | |

| | | |
|---|---|---|
| FIA_AFL.1 | a) The reaching of the threshold for the unsuccessful authentication attempts and the actions (e.g., disabling of an account) taken and the subsequent, if appropriate, restoration to the normal state (e.g., re-enabling of a terminal). | Identity of the unsuccessfully authenticated user. |
| FIA_ATD.1(1) | None. | |
| FIA_ATD.1(2) | None. | |
| FIA_UAU.2 | a) Successful and unsuccessful use of authentication mechanisms;<br>b) All use of the authentication mechanism. | Claimed identity of the user using the authentication mechanism.<br>Success or failure of the authentication mechanism. |
| FIA_UAU_(EXP).5 | a) The final decision on authentication;<br>b) The result of each activated mechanism together with the final decision. | Claimed identity of the user attempting to authenticate. |
| FIA_UID.2 | a) Unsuccessful use of the user identification mechanism, including the user identity provided;<br>b) All use of the user identification mechanism, including the user identity provided (that is, those that authenticate to the TOE). | Claimed identity of the user using the identification mechanism. |

| | | |
|---|---|---|
| FIA_USB.1 | a) Unsuccessful binding of user security attributes to a subject (e.g., creation of a subject).<br>b) Success and failure of binding of user security attributes to a subject. | The identity of the user whose attributes are attempting to be bound. |
| FMT_MOF.1(1) | All modifications in the behavior of the functions in the TSF. | The identity of the administrator performing the function. |
| FMT_MOF.1(2) | a) Enabling or disabling of the key-generation self-tests.<br>b) All modifications in the behavior of the functions in the TSF. | The identity of the administrator performing the function. |
| FMT_MOF.1(3) | All modifications in the behavior of the functions in the TSF. | The identity of the administrator performing the function. |
| FMT_MOF.1(4) | All modifications in the behavior of the functions in the TSF. | The identity of the administrator performing the function. |
| FMT_MOF.1(5) | All modifications in the behavior of the functions in the TSF. | The identity of the administrator performing the function. |
| FMT_MOF.1(6) | All modifications in the behavior of the functions in the TSF. | The identity of the administrator performing the function. |
| FMT_MSA.1(1) | All manipulation of the security attributes. | The identity of the administrator performing the function. |
| FMT_MSA.1(2) | All manipulation of the security attributes. | The identity of the administrator performing the function. |

| | | |
|---|---|---|
| FMT_MSA.3(1) | a) Modifications of the default setting of permissive or restrictive rules;<br>b) All modifications of the initial values of security attributes. | |
| FMT_MSA.3(2) | a) Modifications of the default setting of permissive or restrictive rules;<br>b) All modifications of the initial values of security attributes. | |
| FMT_MTD.1(1) | All modifications of the values of TSF data by the administrator. | The identity of the administrator performing the function. |
| FMT_MTD.1(2) | All modifications of the values of cryptographic security data by the cryptographic administrator. | The identity of the administrator performing the function. |
| FMT_MTD.1(3) | All modifications to the time and date used to form the time stamps by the administrator. | The identity of the administrator performing the function. |
| FMT_MTD.1(4) | All modifications to the information flow policy ruleset by the Security Administrator. | The identity of the security administrator performing the function. |
| FMT_MTD.2(1) | a) All modifications of the limits on TSF data<br>b) All modifications in the actions to be taken in case of violation of the limits. | The identity of the administrator performing the function. |

| | | |
|---|---|---|
| FMT_MTD.2(2) | a) All modifications of the limits on TSF data.<br>b) All modifications in the actions to be taken in case of violation of the limits. | The identity of the administrator performing the function. |
| FMT_REV.1 | a) Unsuccessful revocation of security attributes;<br>b) All attempts to revoke security attributes. | List of security attributes that were attempted to be revoked.<br>The identity of the administrator performing the function. |
| FMT_SMF.1 | Use of the management functions. | |
| FMT_SMR.2 | a) Modifications to the group of users that are part of a role;<br>b) Unsuccessful attempts to use a role due to given conditions on the roles. | User IDs which are associated with the modifications.<br>The identity of the administrator performing the function. |
| FPT_FLS.1 | Failure of the TSF. | |
| FPT_ITA.1 | The absence of TSF data when required by a TOE. | |
| FPT_ITC.1 | None. | |

| | | |
|---|---|---|
| FPT_ITI.1 | a) The detection of modification of transmitted TSF data.<br>b) The action taken upon detection of modification of transmitted TSF data. | |
| FPT_PRO_(EXP).1 | None. | |
| FPT_RCV.2 | a) The fact that a failure or service discountinuity occurred;<br>b) Resumption of the regular operation;<br>c) Type of failure or service discontinuity. | |
| FPT_RPL.1 (including replay of authentication data notification from the authentication server) | Detected replay attacks. | Identity of the user that was the subject of the reply attack |
| FPT_RVM.1 | None. | |
| FPT_SEP.2 | None. | |
| FPT_STM.1 | Changes to the time. | |
| FPT_TST_(EXP).4 | Execution of this set of TSF self tests and the results of the tests. | The identity of the administrator performing the test, if initiated by an administrator. |

| | | |
|---|---|---|
| FPT_TST_(EXP).5 | Execution of this set of TSF self tests and the results of the tests. | The identity of the administrator performing the test, if initiated by an administrator. |
| FRU_RSA.1(1) | a) Rejection of allocation operation due to resource limits.<br>b) All attempted uses of the resource allocation functions for resources that are under control of the TSF. | |
| FRU_RSA.1(2) | a) Rejection of allocation operation due to resource limits.<br>b) All attempted uses of the resource allocation functions for resources that are under control of the TSF. | |
| FTA_SSL.3 | The termination of a remote session by the session locking mechanism. | The identity of the user associated with the session that was terminated. |
| FTA_TAB.1 | None. | |
| FTA_TSE.1 | a) Denial of a session establishment due to the session establishment mechanism.<br>b) All attempts at establishment of a user session. | The identity of the user attempting to establish the session.<br>For unsuccessful attempts, the reason for denial of the establishment attempt. |

| | | |
|---|---|---|
| FTP_ITC.1(1) | a) Failure of the trusted channel functions.<br>b) Identification of the initiator and target of failed trusted channel functions.<br>c) All attempted uses of the trusted channel functions.<br>d) Identifier of the initiator and target of all trusted channel functions. | Identification of the initiator and target of all trusted channels. |
| FTP_ITC.1(2) | a) Failure of the trusted channel functions.<br>b) Identification of the initiator and target of failed trusted channel functions.<br>c) All attempted uses of the trusted channel functions.<br>d) Identifier of the initiator and target of all trusted channel functions. | Identification of the initiator and target of all trusted channels. |
| FTP_TRP.1(1) | a) Failures of the trusted path functions.<br>b) Identification of the user associated with all trusted path failures, if available.<br>c) All attempted uses of the trusted path functions.<br>d) Identification of the user associated with all trusted path invocations, if available. | Identification of the claimed user identity. |

| | | |
|---|---|---|
| FTP_TRP.1(2) | a) Failures of the trusted path functions.<br>b) Identification of the user associated with all trusted path failures, if available.<br>c) All attempted uses of the trusted path functions.<br>d) Identification of the user associated with all trusted path invocations, if available. | Identification of the claimed user identity. |

#### 5.1.1.4 User identity association (FAU_GEN.2-NIAP-0410)

FAU_GEN.2.1-NIAP-0410 - For audit events resulting from actions of identified users, the TSF shall be able to associate each auditable event with the identity of the user that caused the event.

#### 5.1.1.5 Potential violation analysis (FAU_SAA.1-NIAP-0407)

FAU_SAA.1.1-NIAP-0407 – The TSF shall be able to apply a set of rules in monitoring the audited events and based upon these rules indicate a potential violation of the TSP.

FAU_SAA.1.2-NIAP-0407 - **Refinement:** The TSF shall monitor the accumulation or combination of the following events known to indicate a potential security violation:

a) Security Administrator specified number of user authentication failures;

b) Any detected replay of TSF data or security attributes;

c) Any failure of the cryptographic self-tests;

d) Any failure of the other TSF self-tests;

e) Cryptographic Administrator specified number of encryption failures;

f) Cryptographic Administrator specified number of decryption failures; and

g) [selection: [assignment: additional events from the set of defined auditable events], "no additional events"].

74 *Application Note: The intent of this requirement is that an alarm is generated (FAU_ARP.1) once the threshold for an event is met. Once the alarm has been generated it is assumed that the "count" for that event is reset to zero. The Security Administrator settable number of authentication failures in (a) is intended to be the same value as specified in FIA_AFL.1.1. Note that the user authentication failure is distinct from failure to authenticate data packets, e.g. via ESP or AH.*

75 *The failure of TSF self-tests in (d) include failures of FPT_TST_(EXP).4.*

#### 5.1.1.6 Audit review (FAU_SAR.1)

FAU_SAR.1.1 – The TSF shall provide [the Administrators] with the capability to read [all audit data] from the audit records.

FAU_SAR.1.2 – **Refinement:** The TSF shall provide the audit records in a manner suitable for the Administrators to interpret the information.

76 *Application Note: The Administrator is intended to mean any user acting in an administrative role.*

#### 5.1.1.7 Restricted audit review (FAU_SAR.2)

FAU_SAR.2.1 – **Refinement:** The TSF shall prohibit all users read access to the audit records in the audit trail, except the Administrators.

#### 5.1.1.8 Selectable audit review (FAU_SAR.3)

FAU_SAR.3.1 - The TSF shall provide the ability to perform searches and sorting of audit data based on:

a) [user identity;

b) source subject identity;

c) destination subject identity;

d) ranges of one or more: dates, times, user identities, subject service identifiers, or transport layer protocol;

e) rule identity;

f) TOE network interfaces; and

g) [selection: [assignment: other criteria], no additional criteria]].

77 *Application Note: Audit data should be capable of being searched and sorted on all criteria specified in a – g, if applicable (i.e., not all criteria will exist in all audit records). Sorting means to arrange the audit records such that they are "grouped" together for administrative review. For example, the Audit Administrator may want all the audit records for a specified source subject identity or range of source subject identities (e.g., IP source address or range of IP source addresses) presented together to facilitate their audit review. If no additional criteria are provided by the TOE to perform searches or sorting of audit data, the ST author selects "no additional criteria".*

### 5.1.1.9 Audit event selection (FAU_SEL.1-NIAP-0407)

FAU_SEL.1.1-NIAP-0407 - **Refinement:** The TSF shall allow only the Audit Administrator to include or exclude auditable events from the set of audited events based on the following attributes:

a) user identity;

b) event type;

c) [selection: object identity, subject identity, host identity, "none"];

d) success of auditable security events;

e) failure of auditable security events; and

f) [selection: [assignment: list of additional criteria that audit selectivity is based upon], no additional criteria]].

78 *Application Note: "event type" is to be defined by the ST author; the intent is to be able to include or exclude classes of audit events.*

### 5.1.1.10 Protected audit trail storage (FAU_STG.1-NIAP-0429)

FAU_STG.1.1-NIAP-0429 – **Refinement**: The TSF shall **restrict the deletion of stored** audit records in the audit trail **to the** Audit Administrator.

FAU_STG.1.2-NIAP-0429 – The TSF shall be able to *prevent* modifications to the audit records in the audit trail.

#### 5.1.1.11 Action in case of possible audit data loss (FAU_STG.3)

FAU_STG.3.1 - **Refinement**: The TSF shall [immediately alert the administrators by displaying a message at the local console, and at the remote administrative console when an administrative session exists for each of the defined administrative roles, at the option of the Security Administrator generate an audible alarm, [selection: [assignment: other methods], no other methods] if the audit trail exceeds [a Security Administrator settable percentage of storage capacity].

79 *Application Note: As with FAU_ARP.1, the TSF provides a message to the local console regardless of whether an administrator is logged in. The message is displayed at the remote console if an administrator is already logged in, or when an administrator logs in. This requirement specifies that the message is sent to the first established session for each of the defined roles to ensure someone in the administrator staff is aware of the alert as soon as possible.*

#### 5.1.1.12 Site-configurable Prevention of audit data loss (FAU_STG.NIAP-0414-1-NIAP-0429)

FAU_STG.NIAP-0414-1.1-NIAP-0429. The TSF shall provide the Audit Administrator the capability to select one or more of the following actions [*selection: 'ignore auditable events', 'prevent auditable events, except those taken by the authorized user with special rights', 'overwrite the oldest stored audit records'*] and [*assignment: other actions to be taken in case of audit storage failure*] to be taken if the audit trail is full.

FAU_STG.NIAP-0414-1.2-NIAP-0429. **Refinement:** The TSF shall **enforce the Audit Administrator's** [*selection: choose one of: "ignore auditable events", "prevent auditable events, except those taken by the authorised user with special rights", "overwrite the oldest stored audit records"*] and [*assignment: other actions to be taken in case of audit storage failure*] if the audit trail is full.

### 5.1.2 Cryptographic support (FCS)

#### 5.1.2.1 Baseline Cryptographic Module (FCS_BCM_(EXP).1)

FCS_BCM_(EXP).1.1 All cryptographic modules shall comply with FIPS PUB 140-2 when performing FIPS-approved cryptographic functions in FIPS-approved cryptographic modes of operation.

FCS_BCM_(EXP).1.2 Cryptographic functions and cryptographic modes of operation as identified in this PP shall be NSA-validated.

80 *Application Note: In time, PP cryptographic requirements are expected to evolve such that NSA-validated cryptographic modules shall only contain cryptographic functions, cryptographic modes of operation, and other types of cryptographic processing that are compliant with this protection profile.*

FCS_BCM_(EXP).1.3 All cryptographic modules implemented in the TSF [selection:

a) Entirely in hardware shall have a minimum overall rating of FIPS PUB 140-2, Level 3;

b) Entirely in software shall have a minimum overall rating of FIPS PUB 140-2, Level 1 and also meet FIPS PUB 140-2, Level 3 for the following: Cryptographic Module Ports and Interfaces; Roles, Services and Authentication; Cryptographic Key Management; Design Assurance; and FIPS PUB 140-2, Level 4 Self Tests[1] as defined by this Protection Profile;

c) As a combination of hardware and software shall have a minimum overall rating of FIPS PUB 140-2, Level 1 and also meet FIPS PUB 140-2,Level 3 for the following: Cryptographic Module Ports and Interfaces; Roles, Services and Authentication; Cryptographic Key Management; Design Assurance; and FIPS PUB 140-2,Level 4 Self Tests[2] as defined by this Protection Profile.]

### 5.1.2.2 Cryptographic Key Generation (for symmetric keys using RNG) (FCS_CKM.1(1))

FCS_CKM.1.1(1) **Refinement:** The TSF shall generate[3] **symmetric** cryptographic keys in accordance with a specified cryptographic key generation algorithm **as follows**: [selection:

a) **a hardware random number generator (RNG) as specified in FCS_COP_(EXP).1, but with a NIST-approved hashing function required for mixing, and/or**

b) **a software RNG as specified in FCS_COP_(EXP).1, and/or**

---

[1] Security Level 4 Self Tests comprise the Security Level 1 Self Tests in FIPS PUB 140-2 and the Statistical RNG Tests in Appendix G of this protection profile. These Statistical RNG Tests are the same as those included in the 25 May 2001 version of FIPS PUB 140-2.

[2] See previous footnote.

[3] This requirement applies strictly to **generation** of symmetric keys. **Validation** techniques for generated symmetric keys are discussed in FCS_CKM_EXP.1.1.

c) **a key establishment scheme as specified in FCS_COP.1(4) based upon public key cryptography using a software RNG as specified in FCS_COP_(EXP).1, and/or a hardware RNG as specified in FCS_COP_(EXP).1, but with a NIST-approved hashing function required for mixing].**

that meets the following:

a) **All cases: (i.e., any of the above)**
**FIPS PUB 180-2, Secure Hash Algorithm;**

b) **Case: Finite field-based key establishment schemes**
**ANSI X9.42-2001, Public Key Cryptography for the Financial Services Industry: Agreement of Symmetric Keys Using Discrete Logarithm Cryptography;** [4]

81 *Application Note: For example, "Classic" Diffie-Hellman-based schemes*

c) **Case: RSA-based key establishment schemes (with odd e)**
**ANSI X9.31-1998 (May 1998), Digital Signatures Using Reversible Public Key Cryptography for the Financial Services Industry (rDSA) for generation of the RSA;**[5] **and**

82 *Application Note: Although ANSI X9.31 is a standard intended for digital signatures, it is being used here for its coverage of the generation of RSA parameters since ANSI X9.44 is still under development. Once ANSI X9.44 is approved it will be referenced here.*

d) **Case: Elliptic curve-based key establishment schemes**
**ANSI X9.63-200x (1 Oct 2000), Public Key Cryptography for the Financial Services Industry: Key Agreement and Key Transport Using Elliptic Curve Cryptography**. [6]

---

[4] Any pseudorandom RNG used in these schemes for generating private values shall be seeded by a nondeterministic RNG (both types of RNGs meeting RNG requirements in this PP).
[5] A pseudorandom RNG seeded by a nondeterministic RNG (both types of RNGs meeting RNG requirements in this PP) shall be used in the generation of these primes.
[6] Any pseudorandom RNG used in these schemes for generating private values shall be seeded by a nondeterministic RNG (both types of RNGs meeting RNG requirements in this PP).

### 5.1.2.3 Cryptographic Key Generation (for asymmetric keys) (FCS_CKM.1(2))

FCS_CKM.1.1(2) **Refinement:** The TSF shall generate[7] **asymmetric**[8] cryptographic keys in accordance with a **domain parameter generator** and [selection: [9]

a) a random number generator and/or

b) a prime number generator].

that meet the following:

a) Generated key strength shall be equivalent to, or greater than, a symmetric key strength of 128 bits using conservative estimates;

b) ANSI X9.80 (3 January 2000), Prime Number Generation, Primality Testing, and Primality Certificates using random integers with deterministic tests, or constructive generation methods;

c) Case: For domain parameters used in finite field-based key establishment schemes ANSI X9.42-2001, Public Key Cryptography for the Financial Services Industry: Agreement of Symmetric Keys Using Discrete Logarithm Cryptography; [10]

83 *Application Note: For example, "Classic" Diffie-Hellman-based schemes*

---

[7] This requirement applies strictly to **generation** of asymmetric keys. **Validation** techniques for generated asymmetric keys are discussed in FCS_CKM_EXP.1.2.

[8] These are the keys/parameters (e.g., the public/private key pairs) underlying a public key-based key establishment scheme, not the session keys established by such schemes.

[9] A deletion of CC text was performed in FCS_CKM.1.1(2). Rationale: The words "specified cryptographic key generation algorithm " and " and specified cryptographic key sizes [assignment: *cryptographic key sizes*] " were deleted for clarity and better flow of the requirement. The parameters for generating asymmetric keys can be generated by using different criteria. By deleting the CC words, the element better states the intended requirement.

FCS_CKM.1.1(2) - **Refinement:** The TSF shall generate **asymmetric** cryptographic keys in accordance with a **domain parameter generator** and ***[selection:***
***a random number generator and/or***
***a prime number generator]***.
that meet the…

[10] Any pseudorandom RNG used in these schemes for generating private values shall be seeded by a nondeterministic RNG (both types of RNGs meeting RNG requirements in this PP).

d) Case: For domain parameters used in RSA-based key establishment schemes (with odd

e) ANSI X9.31-1998 (May 1998), Digital Signatures Using Reversible Public Key Cryptography for the Financial Services Industry (rDSA) for the generation of the RSA parameters[11]; and

84 *Application Note: Although ANSI X9.31 is a standard intended for digital signatures, it is being used here for its coverage of the generation of RSA parameters since ANSI X9.44 is still under development. Once ANSI X9.44 is approved it will be referenced here.*

f) Case: For domain parameters used in elliptic curve-based key establishment schemes

g) ANSI X9.63-200x (1 Oct 2000), Public Key Cryptography for the Financial Services Industry: Key Agreement and Key Transport using Elliptic Curve Cryptography. [12]

### 5.1.2.4 Cryptographic key distribution (FCS_CKM.2)

FCS_CKM.2.1 The TSF shall distribute cryptographic keys in accordance with a specified cryptographic key distribution method [selection: Manual (Physical) Method, Automated (Electronic) Method, Manual Method and Automated Method] that meets the following:

a) Manual (Physical) Methods:

- The TSF shall support manual distribution of symmetric key in accordance with FIPS PUB 171 (Key Management Using ANSI X9.17).[13]

[11] A pseudorandom RNG seeded by a nondeterministic RNG (both types of RNGs meeting RNG requirements in this PP) shall be used in the generation of these primes.
[12] Any pseudorandom RNG used in these schemes for generating private values shall be seeded by a nondeterministic RNG (both types of RNGs meeting RNG requirements in this PP).
[13] Until NIST identifies approved methods for manually distributing symmetric key, FIPS PUB 171 (Key Management Using ANSI X9.17) shall be used. For purposes of interpreting FIPS PUB 171, only the Triple Data Encryption Algorithm (TDEA) with 168 bits of key shall be applied. (Data Encryption Standard (DES) is not acceptable for meeting this requirement. Eventual migration to AES is expected.)

- The TSF shall support manual distribution of private asymmetric key material (certificates and/or keys) in accordance with NSA-certified DOD PKI for public key distribution using NSA-approved certificate schemes[14] with hardware tokens for protection of private keys that meet the following:

  1) PKI Roadmap for the DoD,

  2) DoD X.509 Certificate Policy,

  3) PKCS #8 v1.2 (Private-Key Information Syntax Standard),

  4) PKCS #12 v1.0 (Personal Information Exchange Syntax),

  5) PKCS #5 v2.0 (Password-Based Encryption Standard, 25 Mar 1999 - Final), and

  6) PKCS #11 v2.11 (Cryptographic Token Interface Standard).

- The TSF shall support manual distribution of public asymmetric key material (certificates and/or keys) in accordance with NSA-certified DOD PKI for public key distribution using NSA-approved certificate schemes[15] for protection of public keys that meet the following:

  1) PKI Roadmap for the DoD,

  2) DoD X.509 Certificate Policy,

  3) PKCS#12 v1.0 (Personal Information Exchange Syntax),

b) Automated (Electronic) Methods:

---

[14] DoD multilevel applications require Class 5 PKI to address worst case environments, but currently this class is just a concept. In the interim, NSA-approved certificate schemes with hardware tokens for protection of private key are approved under the added requirement that stronger protection mechanisms must be applied at the boundaries of the protected environment as stated earlier in this PP. When Class 5 certificates are fully established, they will be required.

[15] See previous footnote.

- The TSF shall automatically distribute symmetric keys in accordance with FIPS PUB 171 (Key Management Using ANSI X9.17).[16].

- The TSF shall automatically distribute public asymmetric key material (certificates and/or keys) in accordance with NSA-certified DoD PKI for public key distribution using NSA-approved certificate schemes[17] that meet the following:

  1) PKI Roadmap for the DoD,

  2) DoD X.509 Certificate Policy,

  3) PKCS#12 v1.0 (Personal Information Exchange Syntax),

- The TSF shall only support manual distribution of private asymmetric key material (certificates and/or keys) in accordance with NSA-certified DOD PKI for public key distribution using NSA-approved certificate schemes[18] with hardware tokens for protection of private keys that meet the following:

  4) PKI Roadmap for the DoD,

  5) DoD X.509 Certificate Policy,

  6) PKCS #8 v1.2 (Private-Key Information Syntax Standard)

  7) PKCS #12 v1.0 (Personal Information Exchange Syntax Standard)

  8) PKCS #5 v2.0 (Password-Based Encryption Standard, 25 Mar 99-Final)and,

---

[16] Until NIST identifies approved methods for automatically distributing symmetric key, FIPS PUB 171 (Key Management Using ANSI X9.17) is being used here. For purposes of interpreting FIPS PUB 171, only TDEA with 168 bits of key shall be applied. (DES is not acceptable for meeting this requirement. Eventual migration to AES is expected.) Where public key schemes are used in key transport methods, NIST Special Publication 800-56 ("Recommendation on Key Establishment Schemes"; DRAFT 2.0, January 2003) shall also be used.

[17] DoD multilevel applications require Class 5 PKI to address worst case environments, but currently this class is just a concept. In the interim, NSA-approved certificate schemes with hardware tokens for protection of private key are approved under the added requirement that stronger protection mechanisms must be applied at the boundaries of the protected environment as stated earlier in this PP. When Class 5 certificates are fully established, they will be required.

[18] See previous footnote.

9) PKCS #11 v2.11 (Cryptographic Token Interface Standard)

### 5.1.2.5 Cryptographic key destruction (FCS_CKM.4)

FCS_CKM.4.1: **Refinement:** The TSF shall destroy cryptographic keys in accordance with a **cryptographic key zeroization method** that meets the following:[19]

a) FIPS PUB 140-2;

b) Zeroization of all plaintext cryptographic keys and all other critical cryptographic security parameters shall be immediate and complete; and

c) For embedded cryptographic modules, the zeroization shall be executed by overwriting the key/critical cryptographic security parameter storage area three or more times using a different alternating data pattern each time.

85 *Application Note: Although verification of this zeroization of a plaintext key/critical cryptographic security parameter is desired here (by checking for the final known alternating data pattern), it is not required at this time. However, vendors are highly encouraged to incorporate this verification whenever possible into their implementations.*

86 *Application Note: Zeroization of any storage, such as memory buffers, that is included in the path of a plaintext key/critical cryptographic security parameter is addressed in FCS_CKM_EXP.2 (Cryptographic Key Handling and Storage).*

### 5.1.2.6 Cryptographic Key Validation and Packaging (FCS_CKM_(EXP).1)

FCS_CKM_(EXP).1.1: The TSF shall apply validation techniques (e.g., parity bits or checkwords) to generated **symmetric** keys in accordance with:

a) FIPS PUB 46-3 (Data Encryption Standard (DES)), and

---

[19] A deletion of CC text was performed in FCS_CKM.4.1. Rationale: The words "specified" and the assignment "[assignment: cryptographic key destruction method]" were deleted because FIPS PUB 140-2 does not provide specific names for the key destruction method.

FCS_CKM.4.1: **Refinement:** The TSF shall destroy cryptographic keys in accordance with a ~~specified~~ **cryptographic key destruction method** ~~[assignment: *cryptographic key destruction method*]~~ that meets …

b) FIPS PUB 171[20] (Key Management Using ANSI X9.17).

FCS_CKM_(EXP).1.2**:** The TSF shall apply validation techniques to generated **asymmetric** keys in accordance with the standards corresponding to the generation technique as called out in FCS_CKM.1.1(2).

FCS_CKM_(EXP).1.3**:** Any public key certificates generated by the TSF shall be in accordance with NSA-certified NSA-approved certificate schemes[21].

87 *Application Note: FIPS 46-3, FIPS 74, and FIPS 81 may be withdrawn according to NIST Docket No. 040602169-4169-01. This docket announces the proposed withdrawal of FIPS for the Data Encryption Standard (DES). Comments are required to be submitted to NIST by 9 September 2004 on why it shouldn't be withdrawn.*

### 5.1.2.7 Cryptographic Key Handling and Storage (FCS_CKM_(EXP).2)

FCS_CKM_(EXP).2.1**:** The TSF shall perform key entry and output in accordance with FIPS PUB 140-2, Level 3.

FCS_CKM_(EXP).2.2**:** The TSF shall provide a means to ensure that keys are associated with the correct entities (i.e., person, group, or process) to which the keys are assigned.

FCS_CKM_(EXP).2.3**:** The TSF shall perform a key error detection check on each transfer of key (internal, intermediate transfers).

88 *Application Note: A parity check is an example of a key error detection check.*

FCS_CKM_(EXP).2.4: The TSF shall encrypt or split persistent secret and private keys when not in use.

---

[20] For purposes of interpreting this standard, only TDEA with 168 bits of key shall be applied (DES is not acceptable for meeting this requirement. Eventual migration to Advanced Encryption Standard (AES) is expected.).

[21] DoD multilevel applications require Class 5 PKI to address worst case environments, but currently this class is just a concept. In the interim, NSA-approved certificate schemes with hardware tokens for protection of private keys are approved under the added requirement that stronger protection mechanisms must be applied at the boundaries of the protected environment as stated earlier in this PP. When Class 5 certificates are fully established, they will be required.

89 *Application Note: A persistent key, such as a file encryption key, is one that must be available in the system over long periods of time. A non-persistent key, such as a key used to encrypt or decrypt a single message or a session, is one that is ephemeral in the system.*

90 *Application Note: "When not in use" shall be interpreted in the strictest sense so that persistent keys only exist in plaintext form during intervals of operational necessity. For example, a file encryption key shall exist in plaintext form only during actual encryption and/or decryption processing of a file. Once the file is decrypted or encrypted the file encryption key shall be immediately covered for protection.*

FCS_CKM_(EXP)_2.5 The TSF shall destroy non-persistent cryptographic keys after an administrator-defined period of time of inactivity.

FCS_CKM_(EXP).2.6: The TSF shall overwrite each intermediate storage area for plaintext key/critical cryptographic security parameter (i.e., any storage, such as memory buffers, that is included in the path of such data). This overwriting shall be executed three or more times using a different alternating data pattern each time upon the transfer of the key/critical cryptographic security parameter to another location.

91 *Application Note: This is related to the elimination of internal, temporary copies of plaintext keys created during processing, not to the total destruction of a key from the TOE which is discussed under Key Destruction. Although verification of the zeroization of each intermediate location of a plaintext key/critical cryptographic security parameter is desired here (by checking for the final known alternating data pattern), it is not required at this time. However, vendors are highly encouraged to incorporate this verification whenever possible into their implementations.*

FCS_CKM_(EXP).2.7: The TSF shall prevent archiving of expired (private) signature keys.

92 *Application Note: This requirement is orthogonal to typical system back-up procedures. Therefore, it does not address the problem of archiving an active (private) signature key during a system back-up and saving the key beyond its intended life span.*

### 5.1.2.8 Cryptographic Operations Availability (FCS_COA_(EXP).1)

FCS_COA_(EXP).1 The TSF shall provide the following cryptographic operations:

a) encryption

b) decryption

c) digital signature

d) secure hashing

e) key agreement

f) [assignment: any other cryptographic operations provided].

### 5.1.2.9 Cryptographic Operation (for data encryption/decryption) (FCS_COP.1(1))

FCS_COP.1.1(1) **Refinement:** The TSF shall perform **data encryption/decryption services** in accordance with a **NIST-approved implementation of the** cryptographic algorithm **Triple Data Encryption Algorithm[22] (TDEA) or Advanced Encryption Algorithm (AES) used in NIST-approved modes of operation** and cryptographic key size of **168 bits (three independent keys)(for TDEA) or 192 or 256 bits for AES** that meets the following:

a) **FIPS PUB 140-2, Security Requirements for Cryptographic Modules,**

b) **FIPS PUB 46-3, Data Encryption Standard, and**

c) ANSI X9.52-1998, Triple Data Encryption Algorithm Modes of Operation or

d) FIPS PUB 197, Advanced Encryption Standard (AES)

93 *Application Note: FIPS 46-3, FIPS 74, and FIPS 81 may be withdrawn according to NIST Docket No. 040602169-4169-01. This docket announces the proposed withdrawal of FIPS for the Data Encryption Standard (DES).*

[22] The Advanced Encryption Standard (AES) employing key lengths of 128 bits or greater and meeting NIST-approved AES standards will be required when AES is fully established. With the approval of FIPS PUB 197 and NIST Special Publication 800-38A, progress is being made to fully establish AES, but establishment is not yet complete. Other approved public standards or NIST special publications are still needed for AES. (An example of this is key distribution for AES.)

### 5.1.2.10 Cryptographic Operation (for cryptographic signature) (FCS_COP.1(2))

FCS_COP.1.1(2) **Refinement:** The TSF shall perform **cryptographic signature services** in accordance with the **NIST-approved digital signature** algorithm [selection]:

a) **Digital Signature Algorithm (DSA) with a key size (modulus) of 2048[23] bits or greater**,

b) **RSA Digital Signature Algorithm (rDSA with odd e) with a key size (modulus) of 2048[24] bits or greater, or**

c) **Elliptic Curve Digital Signature Algorithm (ECDSA) with a key size of 256 bits or greater]**

94 *Application Note: For elliptic curve-based schemes, the key size refers to the log2 of the order of the base point. As the preferred approach for cryptographic signature, elliptic curves will be required within a TBD time frame after all the necessary standards and other supporting information are fully established.*

that meets the following:

a) Case: Digital Signature Algorithm
FIPS PUB 186-2[25], Digital Signature Standard, for signature creation and verification processing; and ANSI Standard X9.42-2001, Public Key Cryptography for the Financial Services Industry: Agreement of Symmetric Keys Using Discrete Logarithm Cryptography for generation of the domain parameters[26];

---

[23] A 2048-bit or greater modulus is required to provide the desired 128-bit equivalent symmetric key strength. The 2048-bit modulus is compatible with (1.) operationally practical digital signature key sizes in pending IPSEC commercial products, and (2.) the current direction of digital signatures in the DoD PKI. This smaller modulus reduces the equivalent symmetric key strength to 112 bits. Certificate signatures based on a 2048-bit or greater modulus or the elliptic curve approach is recommended as soon as the DoD PKI can support it. The elliptic curve approach is preferred. *{"Nearterm applications" means products designed and validated against this specific version of the PP.}*

[24] See previous footnote.

[25] FIPS PUB 186-3 is under development. It will incorporate the signature creation and verification processing of FIPS PUB 186-2, and the generation of domain parameters of ANSI X9.42. FIPS PUB 186-3 shall be used here when it is finalized and approved.

[26] Any pseudorandom RNG used in these schemes for generating private values shall be seeded by a nondeterministic RNG (both types of RNGs meeting RNG requirements in this PP).

b) **Case: RSA Digital Signature Algorithm (with odd e)**
**ANSI X 9.31-1998 (May 1998), Digital Signatures Using Reversible Public Key Cryptography For The Financial Services Industry (rDSA)[27];**

c) **Case: Elliptic Curve Digital Signature Algorithm**
**ANSI X9.62-1-xxxx (10 Oct 1999), Public Key Cryptography for the Financial Services Industry: Elliptic Curve Digital Signature Algorithm (ECDSA)[28].**

#### 5.1.2.11 Cryptographic Operation (for cryptographic hashing) (FCS_COP.1(3))

FCS_COP.1.1(3) **Refinement:** The TSF shall perform **cryptographic hashing services** in accordance with a **NIST-approved hash implementation** of the Secure Hash algorithm and **message digest size of at least 256 bits** that meets the following: **FIPS PUB 180-2**.

95 *Application Note: The message digest size should correspond to double the system encryption key strength.*

#### 5.1.2.12 Cryptographic Operation (for message authentication) (FCS_COP.1(4))

FCS_COP.1.1(4) **Refinement:** The TSF shall perform **message authentication services** in accordance with a **NIST-approved message authentication implementation** of the **message digest size of at least 256 bits** that meets the following: **FIPS 198 (HMAC) or NIST Special Publication 800-38C (Counter with Cipher Block Chaining-Message Authentication Code (CCM)).**

96 *Application Note: The HMAC standard specifies that key size shall be at least half the size of the message digests output.*

---

[27] See previous footnote.
[28] See previous footnote.

### 5.1.2.13 Cryptographic Operation (for cryptographic key agreement) (FCS_COP.1(5))

FCS_COP.1.1(5) **Refinement:** The TSF shall perform **cryptographic key agreement services** in accordance with a **NIST-approved implementation of a key agreement**[29] algorithm [selection]:

a) **Finite Field-based key agreement algorithm and cryptographic key sizes(modulus) of 2048 bits or greater,**

b) **Elliptic Curve-based key agreement algorithm and cryptographic key size of 256 bits or greater]**

97 *Application Note: For elliptic curve-based schemes the key size refers to the log2 of the order of the base point. As the preferred approach for key exchange, elliptic curves will be required once necessary standards and other supporting information are fully established.*

that meets the following:

a) **Case: Finite field-based key agreement schemes**
**ANSI X9.42-2001, Public Key Cryptography for the Financial Services Industry: Agreement of Symmetric Keys Using Discrete Logarithm Cryptography[30];**

98 *Application Note: For example, "Classic" Diffie-Hellman-based schemes*

b) **Case: Elliptic curve-based key agreement schemes**
**ANSI X9.63-200x (1 Oct 2000), Public Key Cryptography for the Financial Services Industry: Key Agreement and Key Transport using Elliptic Curve Cryptography.[31]**

99 *Application Note: Some authentication mechanism on the keying material is recommended. In addition, repeated generation of the same shared secrets should*

---

[29] Until FIPS PUB 140-2 identifies approved key agreement schemes, NIST Special Publication 800-56 ("Recommendation on Key Establishment Schemes", DRAFT 2.0, Jan 2003) shall be used here.
[30] Any pseudorandom RNG used in these schemes for generating private values shall be seeded by a nondeterministic RNG (both types of RNGs meeting RNG requirements in this PP).
[31] See previous footnote.

*be avoided. As an example, the MQV schemes described in the above standards address these issues.*

### 5.1.2.14 (FCS_COP_(EXP).1)

FCS_COP_(EXP).1.1 The TSF shall perform all random number generation (RNG) services in accordance with [selection]:

- multiple independent hardware-generated inputs combined with a mixing function, or

100 *Application Note: A NIST-approved hashing function(i.e., SHA-1) is recommended for the mixing function in hardware based RNGs. If the length of the needed random number exceeds the length of the hash's message digest, then multiple hashes can be used to provide the needed random quantity.*

- multiple independent software-generated inputs combined with a NIST-approved hashing function, or

101 *Application Note: A NIST-approved hashing function is required for the mixing function in software based RNGs. If the length of the needed random number exceeds the length of the hash's message digest, then multiple hashes can be used to provide the needed random quantity.*

- a combination of multiple independent hardware-generated inputs combined with a mixing function and multiple independent software-generated inputs combined with a NIST-approved hashing function.]

that meet the following:

a) FIPS PUB 180-2, when using a NIST-approved hashing function as the mixing function,

b) Documents listed in Appendix H and NIST Special Publication 800-22: A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications;

102 *Application Note: This publication includes some discussion and guidance on randomness and RNG seeding. Successful completion and documentation of these tests during the TOE development helps to demonstrate the random number generator design is rigorous. There exists a NIST toolbox for running these tests. Requirements for acceptable thresholds and sample sizes for use in applying NIST Special Publication 800-22 in the context of this protection profile can be found in Appendix D of this profile.*

c) All the RNG/PRNG self-tests of FIPS PUB 140-2,

d) All statistical RNG tests (as specified in Appendix G) upon demand and upon power-up,

e) The augmented tests, and self-test requirements from this PP: TSF Self Testing, and

f) RNG/PRNG design and test documentation consistent with that required in this PP for other subsystems: Development Documentation (ADV)

FCS_COP_(EXP).1.2 The TSF shall defend against tampering of the random number generation (RNG)/ pseudorandom number generation (PRNG) sources.

103 *Application Note: The RNG/PRNG should be resistant to manipulation or analysis of its sources, or any attempts to predictably influence its states. Three examples of very different approaches the TSF might pursue to address this include: a) identifying the fact that physical security must be applied to the product, b) applying checksums over the sources, or c) designing and implementing the TSF RNG with a concept similar to a keyed hash (e.g., where periodically, the initial state of the hash is changed unpredictably and each change is protected as when provided on a tamper-protected token, or in a secure area of memory.*

### 5.1.2.15 Internet key exchange (FCS_IKE_(EXP).1)

FCS_IKE_(EXP).1.1 The TSF shall provide cryptographic key establishment techniques in accordance with RFC 2409 as follows(s):

Phase 1, the establishment of a secure authenticated channel between the TOE and another remote router endpoint, shall be performed using one of the following, as configured by the security administrator:

- Main Mode
- Aggressive Mode]

New Group mode shall include the private group 14, 2048-bit MOD P, ***[selection:[assignment: other group modes determined by the ST author, ]"no other group modes"]*** for the Diffie-Hellman key exchange.

Phase 2, negotiation of security services for IPsec, shall be done using Quick Mode, using SHA-1 as the pseudo-random function. Quick Mode shall generate key material that provides perfect forward secrecy.

FCS_IKE_(EXP).1.2 The TSF shall require the **nonce, and the x of g^xy** be randomly generated using FIPS-approved random number generator when computation is being performed.

- The recommended nonce sizes are to be between 8 and 256 bytes;
- The minimum size for the x should be 256 bits.

FCS_IKE_(EXP).1.3 When performing authentication using pre-shared keys, the key shall be generated using the FIPS approved random number generator specified in FCS_COP_(EXP).1 .1.

FCS_IKE_(EXP).1.4 The TSF shall compute the value of SKEYID (as defined in RFC 2409), using a NIST-approved hashing function as the pseudo-random function. The TSF shall be capable of authentication using the methods for

- Signatures: SKEYID = sha(Ni_b | Nr_b, g^xy)
- Pre-shared keys: SKEYID = sha(pre-shared-key, Ni_b | Nr_b)
- [*selection: Authentication using Public key encryption, computing SKEYID as follows:* SKEYID = sha(sha(Ni_b | Nr_b), CKY-I | CKY-R), [*assignment: other authentication method],"no other authentication methods"*]

104 *Application Note: If public key encryption is the method of choice, the sha algorithm listed in the requirement will be used. If another option is selected, a different authentication method or a different hash algorithm for generating SKEYID may be specified.*

105 *Refer to RFC 2409 for an explanation of the notation and definitions of the terms.*

FCS_IKE_(EXP).1.5 The TSF shall compute authenticated keying material as follows:

- SKEYID_d = sha(SKEYID, g^xy | CKY-I | CKY-R | 0)
- SKEYID_a = sha(SKEYID, SKEYID_d | g^xy | CKY-I | CKY-R | 1)
- SKEYID_e = sha(SKEYID, SKEYID_a | g^xy | CKY-I | CKY-R | 2)
- [*selection*: [*assignment: other methods for computing the authenticated keying material*], *none*]]

106 *Application Note: If the assignment is selected, a different method for computing the authenticated keying material may be used, or a different hash algorithm may be specified.*

FCS_IKE_(EXP).1.6 To authenticate the Phase 1 exchange, the TSF shall generate HASH_I if it is the initiator, or HASH_R if it is the responder as follows:

- HASH_I = sha(SKEYID, g^xi | g^xr | CKY-I | CKY-R | SAi_b | IDii_b)

- HASH_R = sha(SKEYID, g^xr | g^xi | CKY-R | CKY-I | SAi_b | IDir_b)

107 *Application Note: Refer to RFC 2409 for an explanation of the notation and definitions of the terms.*

FCS_IKE_(EXP).1.7 The TSF shall be capable of authenticating IKE Phase 1 using the following methods as defined in RFC 2409, as configured by the security administrator:

a) **Authentication with digital signatures:** The TSF shall use [selection: RSA, DSA,[selection: [assignment: other digital signature algorithms], "no other digital signature algorithms"]]

b) when an RSA signature is applied to HASH I or HASH R it must be first PKCS#1 encoded. The TSF shall check the HASH_I and HASH_R values sent against a computed value to detect any changes made to the proposed transform negotiated in the phase one. If changes are detected, the session shall be terminated and an alarm shall be generated.

c) [selection:[assignment: X.509 certificates Version 3 [selection: other version of X.509 certificates, "no other versions"]] X.509 V3 implementations, if implemented, shall be capable of checking for validity of the certificate path, and at option of SA, check for certificate revocation using [selection: CRL, OCSP, SVCP].

d) **Authentication with a pre-shared key:** The TSF shall allow authentication using a pre-shared key.

FCS_IKE_(EXP).1.8 The TSF shall compute the hash values for Quick Mode in the following way

HASH(1) = sha(SKEYID_a, M-ID | (assignment: any ISAKMP payload after HASH(1) header contained in the message)

HASH(2) = sha(SKEYID_a, M-ID | Ni_b | (assignment: any ISAKMP payload after HASH(2) header contained in the message)

HASH(3) = sha(SKEYID_a, 0 | M-ID | Ni_b | Nr_b)

108 *Application Note: The following steps will be performed when using the HASH computation:*
*initiator computes HASH(1) and sends to responder*
*responder validates computation of HASH(1) and computes HASH(2) and sends HASH(2) to initiator*
*initiator validates computation of HASH(2) and computes HASH(3) and sends HASH(3) to responder*

109 *IKE is only optional when Security Association (SA) elects not to use perfect forward secrecy.*

110 *Verifying that a TFS implementation actually checks HASH(1) , HASH(2), and HASH(3) values sent against a computed value is important in detecting changes that could have been made to propose transform negotiated in Quick Mode (not as likely as Phase One because Quick Mode is encrypted).*

111 *The ordering of the ISAKMP payloads may differ because Quick Mode only specifies the location of the HASH and SA payload.*

FCS_IKE_(EXP).1.9 The TSF shall compute new keying material during Quick Mode as follows:

[selection: when using perfect forward secrecy

KEYMAT = sha(SKEYID_d, g(qm)^xy | protocol | SPI | Ni_b | Nr_b),

When perfect forward secrecy is not used

KEYMAT = sha(SKEYID_d | protocol | SPI | Ni_b | Nr_b)]

FCS_IKE_(EXP).1.10 The TSF shall at a minimum, support the following ID types: [assignment: ID_IPV4_ADDR, ID_FQDN, ID_USER_FQDN, ID_IPV4_ADDR_SUBNET, ID_IPV6_ADDR, ID_IPV6_ADDR_SUBNET, ID_IPV4_ADDR_RANGE, ID_IPV6_ADDR_RANGE, ID_DER_ASN1_DN, ID_DER_ASN1_GN, ID_KEY_ID].

112 *Application Note: It should be noted that the Internet Protocol Version 6(IPv6) Interim Transistion Guidance memorandum, September 29, 2003, provides support to begin to procure/acquire IPv6 capable GIG assests on October 1, 2003 and a goal for complete transition to IPv6 at FY2008*

### 5.1.3 User Data Protection (FDP)

#### 5.1.3.1 Subset information flow control (unauthenticated policy) (FDP_IFC.1(1) )

FDP_IFC.1.1(1) - The TSF shall enforce the [UNAUTHENTICATED INFORMATION FLOW SFP] on [source subject: TOE interface on which information is received; destination subject: TOE interface to which information is destined; information: network packets; and operations: pass information by opening a relay connection through the TSF on behalf of the source subject to the destination subject, and with the TSF ensuring the following conditions:

a) the connection from the source subject is from a valid peer network,

b) the new relay connection is established to the destination subject on a valid peer network.]

#### 5.1.3.2 Subset information flow control (authenticated policy) (FDP_IFC.1(2))

FDP_IFC.1.1(2) The TSF shall enforce the [AUTHENTICATED INFORMATION FLOW SFP] on [source subject representing authenticated peer routers: source network identifier; destination subject: TOE interface to which information is destined; information: network packets; and operations: pass by opening a relay connection from the TSF on behalf of the source subject to the destination subject, and with the TSF ensuring the following conditions:

a) the connection from the source subject is from a valid peer network,

b) the new relay connection is established to the destination subject on a valid peer network.]

#### 5.1.3.3 Simple Security attributes (unauthenticated policy) (FDP_IFF.1(1))

FDP_IFF.1.1(1) - The TSF shall enforce the [UNAUTHENTICATED INFORMATION FLOW SFP] based on the following types of subject and information security attributes:

a) [Source subject security attributes:

- set of source entity identifiers; and
- [selection: [assignment: other subject security attributes], none].

b) Destination subject security attributes:

- Set of destination entity identifiers; and
- [selection: [assignment: other subject security attributes], none].

113 *Application Note: For the entities, the administrator knows the set of identifiers that can be associated with the physical router interfaces; therefore, they are not "presumed" identifiers. The term "identifiers" was used instead of "addresses" to allow for technologies that are not address-based (e.g., circuit identifiers instead of source and destination addresses).*

114 *The ST author should specify other attributes that are used to identify the source and destination entity sets, based on the technology implemented by the TOE.*

c) Information security attributes:

- presumed identity of source entity[32];
- identity of destination entity;
- transport layer protocol;
- source entity service identifier;
- destination entity service identifier (e.g., TCP or User Datagram Protocol (UDP) destination port number);

115 *Application Note: The transport layer protocol is what is specified in the 8-bit protocol field in the IP header (e.g., this would include ICMP and is not limited to TCP or UDP). The concept of a "service identifier" may differ depending on the networking stack used; the intent is to specify a service that is above the network and transport layers in the protocol stack. A "service" in the IP stack would be NTP, Trivial File Transfer Protocol (TFTP), etc.*

116 *Application Note: Not all of the above security attributes will exist in all network packets. However, the TOE's ruleset allows the Security Administrator to select and filter on any of the above security attributes as part of the policy decision. The intent is that if a network packet includes any of the above security attributes, those attributes may be used in the policy decision. The ST author should fill in the assignment all attributes that the Security Administrator is able to specify when creating the router rules.*

for non-IP-based network stacks: [assignment: information security attributes]].

117 *Application Note: If a compliant TOE uses an IP based network stack (including IP running on top of another protocol, such as Asynchronous Transfer Mode (ATM)), then the first selection is made. If the TOE uses a network stack that is not IP-based (e.g., ATM without IP) then the ST author uses the second selection and fills in the assignment with the attributes that provide a commensurate level of confidence for the protocols employed that network packets can be correctly associated with a connection.*

---

[32] The TOE can make no claim as to the real identity of any source entity; the TOE can only suppose that such identities are accurate. Therefore, a "presumed identity" is used to identify source entities. Note, however, that the TOE can ensure that the identity is included in the set that is associated with the interface (see FDP_IFF.1.6(1)).

FDP_IFF.1.2(1) - **Refinement**: The TSF shall permit an information flow between a **source** entity and a **destination entity** via a controlled operation if the following rules hold:

- [the presumed identity of the source entity is in the set of source entity identifiers;
- the identity of the destination entity is in the set of destination entity identifiers;
- the information security attributes match the attributes in an information flow policy rule (contained in the information flow policy ruleset defined by the Security Administrator) according to the following algorithm [assignment: algorithm used by the TOE to match information security attributes to information flow policy rules]; and
- the selected information flow policy rule specifies that the information flow is to be permitted].

118 *Application Note: In a router, the administrator specifies information flow policy rules that contain information security attribute values (or wildcards that "stand" for multiple values of the same type; e.g., 127.*.*.* would represent any IP address that begins with "127"), and associate with that rule an action that permits the information flow or disallows the information flow. When a packet arrives at the source interface, the information security attribute values of the packet are compared to each information flow policy rule by some TOE-specified algorithm, and when a match is found the action specified by that rule is taken. Since wildcards would allow the specific attributes in a packet to potentially match more than one rule, the ST author needs to fill in the assignment with the algorithm the TOE uses to find a matching rule. This could be "first match", "most specific match", or some more elaborate description.*

FDP_IFF.1.3(1) - The TSF shall enforce the [assignment: additional information flow control SFP rules].

FDP_IFF.1.4(1) - The TSF shall provide the following [the Security Administrator shall have the capability to view all information flows allowed by the information flow policy ruleset before the ruleset is applied].

119 *Application Note: Some routers create additional rules as a side-effect of creating a rule; for example, a router may create a rule allowing a File Transfer Protocol (FTP) data channel when a rule allowing FTP (control connections) is created. This requirement allows an administrator to view the entire ruleset so that they can identify such rules and confirm that the ruleset reflects the desired policy.*

120 *"before the rule set is applied" means that the administrator is able to view the entire rule set before it is put into use on the TOE. This gives the administrator the opportunity to address any errors or unintended flows.*

FDP_IFF.1.5(1) - The TSF shall explicitly authorize an information flow based on the following rules: [none].

FDP_IFF.1.6(1) - The TSF shall explicitly deny an information flow based on the following rules:

- [The TOE shall reject requests for access or services where the presumed source identity of the information received by the TOE is not included in the set of source identifiers for the source subject;

121 *Application Note: The intent of this requirement is to ensure that a user cannot send packets originating on one TOE interface claiming to originate on another TOE interface.*

- The TOE shall reject requests for access or services where the presumed source identity of the information received by the TOE specifies a broadcast identity;

122 *Application Note: A broadcast identity is one that specifies more than one host address on a network. It is understood that the TOE can only know the sub-netting configuration of networks directly connected to the TOE's interfaces and therefore can only be aware of broadcast addresses on those networks.*

- The TOE shall reject requests for access or services where the presumed source identity of the information received by the TOE specifies a loopback identifier;

- The TOE shall reject requests in which the information received by the TOE contains the route (set of host network identifiers) by which information shall flow from the source subject to the destination subject].

### 5.1.3.4 Simple security attributes (authenticated policy) (FDP_IFF.1(2))

FDP_IFF.1.1(2) - The TSF shall enforce the [AUTHENTICATED INFORMATION FLOW SFP] based on the following types of subject and information security attributes:

a) [Source subject security attributes:

- source network identifier; and

- [selection: [assignment: other subject security attributes], none].

123 *Application Note: Note that the above does not require a "userid" (distinct from the network identifier for the subject) to be specified for use in information flow policy rule; if the TOE provides this feature then it should be listed under the second bullet.*

b) Destination subject security attributes:

- Set of destination network identifiers; and
- [selection: [assignment: other subject security attributes], none].

124 *Application Note: The Security Administrator knows the set of identifiers that can be associated with the physical router interfaces; therefore, they are not "presumed" identifiers. The term "identifiers" was used instead of "addresses" to allow for technologies that are not address-based (e.g., circuit identifiers instead of source and destination addresses).*

125 *The ST author should specify other attributes that are used to identify the source subject and destination subject set, based on the technology implemented by the TOE.*

c) Information security attributes:

- identity of source subject;
- identity of destination subject;
- transport layer protocol;
- destination subject service identifier (e.g., TCP destination port number);

126 *Application Note: The concept of a "service identifier" may differ depending on the networking stack used; the intent is to specify a service that is above the network and transport layers in the protocol stack.*

- [selection: [assignment: other information security attributes], none].

127 *Application Note: The ST author should fill in the assignment with all attributes that the administrator is able to specify when creating the router rules.*

for non-IP-based network stacks: [assignment: information security attributes]].

FDP_IFF.1.2(2) - **Refinement**: The TSF shall permit an information flow between a **source** subject and a **destination subject** via a controlled operation if the following rules hold:

- [the source subject has successfully authenticated to the TOE;

- the identity of the destination subject is in the set of destination identifiers;
- the information security attributes match the attributes in a information flow policy rule (contained in the information flow policy ruleset defined by the administrator) according to the following algorithm [assignment: algorithm used by the TOE to match information security attributes to information flow policy rules]; and
- the selected information flow policy rule specifies that the information flow is to be permitted].

128 *Application Note: In a router, the administrator specifies information flow policy rules that contain information security attribute values (or wildcards that "stand" for multiple values of the same type; e.g., 127.*.*.* would represent any IP address that begins with "127"), and associated with that rule an action that permits the information flow or disallows the information flow. When a packet arrives at the source interface, the information security attribute values of the packet are compared to each information flow policy rule by some TOE-specified algorithm, and when a match is found the action specified by that rule is taken. Since wildcards would allow the specific attributes in a packet to potentially match more than one rule, the ST author needs to fill in the assignment with the algorithm the TOE uses to find a matching rule. This could be "first match", "most specific match", or some more elaborate description.*

FDP_IFF.1.3(2) - The TSF shall enforce the [assignment: additional information flow control SFP rules].

FDP_IFF.1.4(2) - The TSF shall provide the following

- [the Security Administrator shall have the capability to view all information flows allowed by the information flow policy ruleset before the ruleset is applied].

129 *Application Note: Some routers create additional rules as a side-effect of creating a rule; for example, a router may create a rule allowing an FTP data channel when a rule allowing FTP (control connections) is created. This requirement allows an administrator to view the entire ruleset so that they can identify such rules and confirm that the ruleset reflects the desired policy.*

130 *"before the ruleset is applied" means that the administrator is able to view the entire rule set before it is put into use on the TOE. This gives the administrator the opportunity to address any errors or unintended flows.*

FDP_IFF.1.5(2) - The TSF shall explicitly authorize an information flow based on the following rules: [none].

FDP_IFF.1.6(2) - The TSF shall explicitly deny an information flow based on the following rules: [none].

131 *Application Note: Note that the checks done in FDP_IFF.1.6(1) do not need to be re-specified because those checks should occur prior to the user authenticating themselves to the router, and thus are part of the unauthenticated policy rather than the authenticated policy.*

#### 5.1.3.5 Full residual information protection (FDP_RIP.2)

FDP_RIP.2.1 The TSF shall ensure that any previous information content of a resource is made unavailable upon the [selection: allocation of the resource to, deallocation of the resource from] all objects.

### 5.1.4 Identification and Authentication (FIA)

TOE security functions implemented by a probabilistic or permutational mechanism (e.g., password or hash function) are required (at EAL2 and higher) to include a strength of function claim. Strength of Function shall be demonstrated for the authentication mechanism used by the administrators to be SOF-medium, as defined in Part 1 of the CC. Specifically, the local authentication mechanism must demonstrate adequate protection against attackers possessing a moderate attack potential.

#### 5.1.4.1 Authentication failure handling (FIA_AFL.1)

FIA_AFL.1.1 The TSF shall detect when *a Security Administrator configurable positive integer within [a Security Administrator configurable amount of time]* unsuccessful authentication attempts occur related to [a user's authentication].

FIA_AFL.1.2 When the defined number of unsuccessful authentication attempts has been met or surpassed, the TSF shall [lock the device for a Security Administrator configurable amount of time].

132 *Application Note: At least one account should be exempted from the FIA_AFL.1.2 requirement in order to prevent denial of access.*

133 *Interp note: This requirement is modified as per CCIMB Interp #111*

#### 5.1.4.2 User attribute definition (Human User Identity) (FIA_ATD.1(1))

FIA_ATD.1.1(1) – **Refinement**: The TSF shall maintain the following list of security attributes belonging to an **authorized** user:

a) [user identifier(s):

role;

[selection: [assignment: Any security attributes related to a user identifier (e.g., certificate associated with the userid)], none]; and

b) [selection: [assignment: other user security attributes], none]].

134 *Application Note: This requirement applies to authorized users: administrators and authorized IT entitiess. The intent is to allow multiple userids to be associated with a user. This allows a single human user to assume multiple roles, albeit requiring authentication as the userid associated with a given role. The intent is for a userid to only be associated with a single role, thus limiting the amount of damage if an administrative role is compromised.*

135 *If a particular TOE has different attributes for administrators and authenticated IT entities, the ST author is expected to iterate this requirement once for each set of users to reflect the differences.*

136 *Item "b" could be used by an ST author to specify a list of the session establishment criteria identified in FTA_TSE depending on the TOE's implementation of the session establishment function.*

### 5.1.4.3 User Attribute definition (TOE to TOE Identification) (FIA_ATD.1(2))

FIA_ATD.1.1(2) **Refinement:** The TSF shall maintain the following list of security attributes belonging to **authorized subjects**:

a) [subject identity;

b) [assignment: any other security attributes].

### 5.1.4.4 User authentication before any action (FIA_UAU.2)

FIA_UAU.2.1 The TSF shall require each user to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that user.

### 5.1.4.5 Authentication Mechanism (FIA_UAU_(EXP).5)

FIA_UAU_(EXP).5.1 - The TSF shall provide a local authentication mechanism, [selection: [assignment: other authentication mechanism(s)], none] to perform user authentication.

137 *Application Note: This explicit requirement is needed because there is no CC requirement (other than FIA_UAU.5) that requires the TSF provide authentication (it is implied by other FIA_UAU requirements, but not explicitly required).*

138 *The ST author could chose to fill in the assignment with any additional authentication mechanism such as a single-use authentication mechanism, or a mechanism that authenticates users by using a certificate. If an asymmetric algorithm is chosen, the TOE may rely upon a certificate authority server to obtain a user's certificate, and this server would be considered an authorized IT entity and IT environment requirements should be levied on this IT entity.*

#### 5.1.4.6 User identification before any action (FIA_UID.2)

FIA_UID.2.1 The TSF shall require each user to identify itself before allowing any other TSFmediated actions on behalf of that user.

#### 5.1.4.7 User-subject binding (FIA_USB.1)

FIA_USB.1.1: The TSF shall associate the following user security attributes with subjects acting on the behalf of that user: [assignment: list of user security attributes].

FIA_USB.1.2: The TSF shall enforce the following rules on the initial association of user security attributes with subjects acting on the behalf of users: [assignment: rules for the initial association of attributes].

FIA_USB.1.3: The TSF shall enforce the following rules governing changes to the user security attributes associated with subjects acting on the behalf of users: [assignment: rules for the changing of attributes].

139 *Application Note: User security attributes are defined in FIA_ATD.1*

140 *Interp note: This requirement is modified as per CCIMB Interp #137.*

### 5.1.5 Security Management (FMT)

#### 5.1.5.1 Management of security functions behavior (TSF non-Cryptographic Self-test) (FMT_MOF.1(1))

FMT_MOF.1.1(1) - The TSF shall restrict the ability to modify the behavior of the functions

[TSF Self-Test (FPT_TST_(EXP).4)]

to [the Security Administrator].

141 *Application Note: "Modify the behavior" refers to specifying the interval at which the test periodically run, or perhaps selecting a subset of the tests to run.*

#### 5.1.5.2 Management of security functions behavior (Cryptographic Self-test) (FMT_MOF.1(2))

FMT_MOF.1.1(2) - The TSF shall restrict the ability to enable, disable the functions

[TSF Self-Test (FPT_TST_(EXP).5)]

to [the Cryptographic Administrator].

142 *Application Note: The enabling or disabling of the cryptographic self-tests immediately after key generation.*

#### 5.1.5.3 Management of security functions behavior (Audit review) (FMT_MOF.1(3))

FMT_MOF.1.1(3) - The TSF shall restrict the ability to enable, disable, determine and modify the behavior of the functions

[Security Audit (FAU_SAR.1, FAU_SAR.2, FAU_SAR.3)] to [an Administrator].

#### 5.1.5.4 Management of security functions behavior (Audit selection) (FMT_MOF.1(4))

FMT_MOF.1.1(4) - The TSF shall restrict the ability to enable, disable, determine and modify the behavior of the functions

[Security Audit Analysis (FAU_SAA); and

Security Audit (FAU_SEL)]

to [the Security Administrator].

143 *Application Note: For the Audit function, enable and disable refer to the ability to enable or disable the audit mechanism as a whole. "Determine the behavior" means the ability to determine specifically what on the system is being audited, while "modify the behavior" means the ability to set or unset specific aspects of the audit mechanism, such as what user behavior is audited, etc.*

#### 5.1.5.5 Management of security functions behavior (Alarms) (FMT_MOF.1(5))

FMT_MOF.1.1(5) - The TSF shall restrict the ability to enable, or disable the functions

[Security Alarms (FAU_ARP)]

to [the Security Administrator].

144 *Application Note: This requirement ensures only the Security Administrator can enable or disable (turn on or turn off) the alarm notification function – messages and/or the audible alarm. As currently written, FAU_ARP.1 does not lend itself to behavior modification. If the ST author were to include additional functionality in FAU_ARP.1 (e.g., notify the administrator via a pager) then the ST author should consider adding, "modify the behavior" to this requirement.*

#### 5.1.5.6 Management of security functions behavior (quota mechanism) (FMT_MOF.1(6))

FMT_MOF.1.1(6) - The TSF shall restrict the ability to determine the behavior of the functions

[Controlled connection-oriented resource allocation (FRU_RSA.1(2));

an administrator-specified network identifier;

set of administrator-specified network identifiers;

administrator-specified period of time]

to [the Security Administrator].

145 *Application Note: "determine the behavior of" refers to specifying the network identifier(s) that will be tracked using the FRU_RSA.1(2) requirement and the time period over which the quota limitations are enforced. Note that the specification of the actual quotas, while part of the resource allocation functionality, is done by FMT_MTD.2(2).*

#### 5.1.5.7 Management of security attributes (unauthenticated) (FMT_MSA.1(1))

FMT_MSA.1.1 The TSF shall enforce the [UNAUTHENTICATED INFORMATION FLOW SFP] to restrict the ability to [selection: change_default, query, modify, delete, [assignment: other operations]] the security attributes [assignment: list of security attributes] to [assignment: the authorized identified roles].

#### 5.1.5.8 Management of security attributes (authenticated) (FMT_MSA.1(2))

FMT_MSA.1.1 The TSF shall enforce the [AUTHENTICATED INFORMATION FLOW SFP] to restrict the ability to [selection: change_default, query, modify, delete, [assignment: other operations]] the security attributes [assignment: list of security attributes] to [assignment: the authorized identified roles].

#### 5.1.5.9 FMT_MSA.3(1)

FMT_MSA.3.1(1) –The TSF shall enforce the [UNAUTHENTICATED INFORMATION FLOW SFP] to provide *restrictive* default values for security attributes that are used to enforce the SFP.

146 *Application Note: "restrictive" in this case means that by default information is not allowed to flow (according to the referenced policies) unless an explicit enforcing attribute allows an information flow. By default, information is not allowed to flow.*

FMT_MSA.3.2(1) - The TSF shall allow the [Security Administrator] to specify alternative initial values to override the default values when an object or information is created.

#### 5.1.5.10 FMT_MSA.3(2)

FMT_MSA.3.1(2) – The TSF shall enforce the [AUTHENTICATED INFORMATION FLOW SFP] to provide *restrictive* default values for security attributes that are used to enforce the SFP.

147 *Application Note: "restrictive" in this case means that by default information is not allowed to flow (according to the referenced policies) unless an explicit enforcing attribute allows an information flow. By default, information is not allowed to flow.*

FMT_MSA.3.2(2) - The TSF shall allow the [Security Administrator] to specify alternative initial values to override the default values when an object or information is created.

#### 5.1.5.11 Management of TSF data (non-cryptographic, non-time TSF data) (FMT_MTD.1(1))

FMT_MTD.1.1(1) – **Refinement:** The TSF shall restrict the ability to [selection: change default, query, modify, delete, clear, [selection: [assignment: other operations], none]] all the [TSF data except cryptographic security data and the time and date

used to form the time stamps in FPT_STM.1] to [the administrators or authorized IT entities].

148 *Application Note: The ST author should iterate this requirement as necessary to ensure that the TSF data are characterized in terms of the functionality provided by the TOE, and that the access is appropriately restricted to the appropriate administrators and authorized IT entities. The cryptographic security data and time stamp data are covered in the following two components, as they have specific requirements to address the PP's threats and policies.*

#### 5.1.5.12 Management of TSF data (cryptographic TSF data) (FMT_MTD.1(2))

FMT_MTD.1.1(2) - The TSF shall restrict the ability to modify the [cryptographic security data] to [the Cryptographic Administrator].

149 *Application Note: The intent of this requirement is to restrict the ability to configure the TOE's cryptographic policy to the Cryptographic Administrator. Configuring the cryptographic policy is related to things such as: setting modes of operation, key lifetimes, selecting a specific algorithm, and key length.*

#### 5.1.5.13 Management of TSF data (time TSF data) (FMT_MTD.1(3))

FMT_MTD.1.1(3) – The TSF shall restrict the ability to [set] the [time and date used to form the time stamps in FPT_STM.1] to [the Security Administrator or authorized IT entity].

150 *Application Note: The ST author is able to restrict the ability to set the time and date to just the Security Administrator, to just an authorized IT entity, or both.*

#### 5.1.5.14 Management of TSF data (Information flow policy ruleset) (FMT_MTD.1(4))

FMT_MTD.1.1(4) – The TSF shall restrict the ability to query, modify, delete, create, [selection: [assignment: other operations], none] the [information flow policy rules] to [the Security Administrator].

151 *Application Note: This restricts the specification of the information flow policy ruleset identified in the FDP_IFF requirements to the Security Administrator. This specification is done using the attributes defined for those policies.*

152 *The ST author should fill in any TOE-specific operations that an administrator can perform on the ruleset in the assignment.*

#### 5.1.5.15 Management of limits on TSF data (transport-layer quotas) (FMT_MTD.2(1))

FMT_MTD.2.1(1) - The TSF shall restrict the specification of the limits for [quotas on transport-layer connections] to [the Security Administrator].

FMT_MTD.2.2(1) - The TSF shall take the following actions, if the TSF data are at, or exceed, the indicated limits: [assignment: actions to be taken].

153 *Application Note: Note that the wording of FRU_RSA.1(1) does not indicate that the TOE must provide the Security Administrator the means to adjust the maximum quota; however, if the TOE does provide such a mechanism then FMT_MTD.2.1(1) would require that that mechanism is restricted to the Security Administrator.*

154 *For FMT_MTD.2.2(1), the ST author should specify the actions that the TOE takes when quota is reached. For the TCP SYN attack, for example, the action may be to drop the oldest "n" half-open connections.*

#### 5.1.5.16 Management of limits on TSF data (controlled connection-oriented quotas) (FMT_MTD.2(2))

FMT_MTD.2.1(2) - The TSF shall restrict the specification of the limits for [quotas on controlled connection-oriented resources] to [the Security Administrator].

FMT_MTD.2.2(2) - The TSF shall take the following actions, if the TSF data are at, or exceed, the indicated limits: [assignment: actions to be taken].

155 *Application Note: For FMT_MTD.2.2(2), the ST author should specify the actions that the TOE takes for each controlled connection-oriented resource when the quota (with respect the specific network identifier or set of network identifiers) established by the Security Administrator is reached. This requirement may have to be iterated to be consistent with FRU_RSA.1(2). See the application note on FRU_RSA.1(2) for more detail on the requirements for the quota mechanism.*

#### 5.1.5.17 Revocation (FMT_REV.1)

FMT_REV.1.1 The TSF shall restrict the ability to revoke security attributes associated with the [selection: users, subjects, objects, [assignment: other additional resources]] within the TSC to [assignment: the authorized identified roles].

FMT_REV.1.2 The TSF shall enforce the rules [assignment: specification of revocation rules].

156 *Interp note: This requirement is modified as per CCIMB Interp #201*

#### 5.1.5.18 Specification of Management Functions (FMT_SMF.1)

FMT_SMF.1.1 The TSF shall be capable of performing the following security management functions: [assignment:

- TSF non-Cryptographic Self-test
- Cryptographic Self-test
- Audit Alarms
- Quota Mechanism].

#### 5.1.5.19 Restrictions on security roles (FMT_SMR.2)

FMT_SMR.2.1 The TSF shall maintain the roles: [assignment:

- Security administrator role,
- Audit administrator role,
- Crypto administrator role].

FMT_SMR.2.2 The TSF shall be able to associate users with roles.

FMT_SMR.2.3 The TSF shall ensure that the conditions [assignment: conditions for the different roles] are satisfied.

157 *Application Note: The administering of the TOE is limited to the capabilities associated with an administrative role.*

### 5.1.6 Protection of the TSF (FPT)

#### 5.1.6.1 Failure with preservation of secure state (FPT_FLS.1)

FPT_FLS.1.1 The TSF shall preserve a secure state when the following types of failures occur: [assignment: all failures].

#### 5.1.6.2 Inter-TSF availability within a defined availability metric (FPT_ITA.1)

FPT_ITA.1.1 The TSF shall ensure the availability of [assignment: list of types of TSF data] provided to a remote trusted IT product within [assignment: a defined availability metric] given the following conditions [assignment: conditions to ensure availability].

158 *Application Note: The router must be sure to follow specifications for the different routing protocols used (e.g., BGP, OSPF, etc.) for the retransmission and avalibilty of routing data to peer routers.*

### 5.1.6.3 Inter-TSF confidentiality during transmission (FPT_ITC.1)

FPT_ITC.1.1 The TSF shall protect all TSF data transmitted from the TSF to a remote trusted IT product from unauthorized disclosure during transmission.

159 *Application Note: This is to protect routing and routing related updates between two routers. An example would be the use of Internet Protocol Version 6 (IPv6) and Internet Protocol Security (IPSEC) so that a router could encrypt routing table communications and authentication with a peer router. This would be a method to protect the confidentiality of the network resources.*

### 5.1.6.4 Inter-TSF detection of modification (FPT_ITI.1)

FPT_ITI.1.1 The TSF shall provide the capability to detect modification of all TSF data during transmission between the TSF and a remote trusted IT product within the following metric: [assignment: a defined modification metric].

FPT_ITI.1.2 The TSF shall provide the capability to verify the integrity of all TSF data transmitted between the TSF and a remote trusted IT product and perform [assignment: action to be taken] if modifications are detected.

160 *Application Note: This is to protect routing and routing related updates between two routers. An example would be the use of Internet Protocol Version 6 (IPv6) and Internet Protocol Security (IPSEC) so that a router could protect and detect modifaction of peer communications by encrypting routing table communication and authentication with a peer router.*

### 5.1.6.5 Standard Protocol Usage (FPT_PRO_(EXP).1)

FPT_PRO_(EXP).1 The TSF shall utilize the standard protocol mechanisms within the standard protocols

[selection:[assignment:

a) BGP

b) list of standard protocols]].

#### 5.1.6.6 Automated Recovery (FPT_RCV.2)

FPT_RCV.2.1 When automated recovery from [assignment: list of failures/service discontinuities] is not possible, the TSF shall enter a maintenance mode where the ability to return to a secure state is provided.

FPT_RCV.2.2 For [assignment: list of failures/service discontinuities], the TSF shall ensure the return of the TOE to a secure state using automated procedures.

#### 5.1.6.7 Replay detection (FPT_RPL.1)

FPT_RPL.1.1 - The TSF shall detect replay for the following entities: [TSF data and security attributes].

FPT_RPL.1.2 - The TSF shall perform

[reject data;

audit event; and

[selection: [assignment: list of specific actions], none]]

when replay is detected.

161 *Application Note: Receiving multiple network packets due to network congestion or lost packet acknowledgments is not considered a replay attack. The intent of this requirement is to ensure that an administrative session (in part, in its entirety, by a remote administrator or an authorized IT entity) or a user's authentication sequence cannot be replayed.*

#### 5.1.6.8 Non-bypassability of the TSP (FPT_RVM.1)

FPT_RVM.1.1 The TSF shall ensure that TSP enforcement functions are invoked and succeed before each function within the TSC is allowed to proceed.

#### 5.1.6.9 SFP domain separation (FPT_SEP.2)

FPT_SEP.2.1 - The unisolated portion of the TSF shall maintain a security domain for its own execution that protects it from interference and tampering by untrusted subjects.

FPT_SEP.2.2 - The TSF shall enforce separation between the security domains of subjects in the TSC.

FPT_SEP.2.3 - **Refinement:** The TSF shall maintain the part of the TSF related to [cryptography] in **an address space** for **its** own execution that protects **it** from interference and tampering by the remainder of the TSF and by subjects untrusted with respect to **the cryptographic functionality**.

162 *Application Note: The address space protection would be only for accidental interference (e.g., coding errors) but not from any malicious part of the kernel. It does protect against malicious untrusted subjects. Off board hardware or a third processor hardware state is a preferred implementation, because it would protect the cryptography from all other parts of the TSF. Cryptographic functionality is implemented in cryptomodules and by other code residing in the TSF that has not been validated through the FIPS 140-2 process. All cryptographic functionality, whether implemented in a cryptomodule or in some other way, is covered by the third element of this component.*

### 5.1.6.10 Reliable time stamps (FPT_STM.1)

FPT_STM.1.1 The TSF shall be able to provide reliable time stamps for its own use.

### 5.1.6.11 Inter-TSF basic TSF data consistency (FPT_TDC.1)

FPT_TDC.1.1 The TSF shall provide the capability to consistently interpret [assignment: list of TSF data types] when shared between the TSF and another trusted IT product.

FPT_TDC.1.2 The TSF shall use [assignment: list of interpretation rules to be applied by the TSF] when interpreting the TSF data from another trusted IT product.

### 5.1.6.12 TSF testing (with cryptographic integrity verification) (FPT_TST_(EXP).4)

FPT_TST_(EXP).4.1 –The TSF shall run a suite of self-tests during initial start-up, periodically during normal operation as specified by the Security Administrator, and at the request of an administrator to demonstrate the correct operation of the hardware portions of the TSF.

FPT_TST_(EXP).4.2 –The TSF shall provide an administrator with the capability to use a TSF-provided cryptographic function to verify the integrity of all TSF data except the following: audit data, [selection: [assignment: other dynamic TSF data for which no integrity validation is justified], none].

FPT_TST_(EXP).4.3 - The TSF shall provide an administrator with the capability to use a TSF-provided cryptographic function to verify the integrity of stored TSF executable code.

FPT_TST_(EXP).4.4 - The TSF shall restrict the ability to invoke the self-tests to an Administrator.

163 *Application Note: This explicit requirement is necessary since some TOE data are dynamic (e.g., data in the audit trail, passwords) and so interpretation of "integrity" for FPT_TST.1.2 is required, leading to potential inconsistencies. The intention is that any parameter that only an administrator can control is verified to ensure its integrity is maintained. It is not necessary for the TOE to verify the integrity of audit data or user's passwords. If the TOE verifies the integrity of these, the ST author may fill in the assignment to include them.*

164 *Since this TOE includes all the hardware necessary for the operation of the TOE, the element FPT_TST_(EXP).4.1 ensures that the hardware aspects of the TOE are tested prior to or during operations. It is not necessary to test the software portions of the TSF, since the evaluation ensures the correct operation of the software, software does not degrade or suffer intermittent faults, as does hardware, and integrity of the software portions of the TSF are addressed by FPT_TST_(EXP).4.3. Note that since cryptographic functions implemented in hardware that are part of a cryptomodule are tested in FPT_TST_(EXP).5, this requirement only applies to cryptographic functionality implemented in hardware that is not implemented in a cryptomodule (for instance, an implementation of a Key Agreement algorithm).*

165 *In element 4.2, the ST author should specify the TSF data for which integrity validation is not required, and also specify the administrative role that is able to invoke the integrity verification process. While some TSF data are dynamic and therefore not amenable to integrity verification, it is expected that all TSF data for which integrity verification "makes sense" be subject to this requirement.*

166 *In elements 4.2 and 4.3, the cryptographic mechanism can be any one of the ones specified in FCS_COP.1(2) or FCS_COP.1(3),or FCS_COP.1(4) although typically hash functions or digital signatures are used for integrity verification.*

### 5.1.6.13 Cryptographic self-test (FPT_TST_(EXP).5)

FPT_TST_(EXP).5.1 – The TSF shall run the suite of self-tests provided by the FIPS 140-2 cryptographic module during initial start-up (power on), at the request of an administrator, periodically (at a Security Administrator-specified interval not less than at least once a day) to demonstrate the correct operation of the cryptographic components of the TSF.

FPT_TST_(EXP).5.2 – The TSF shall be able to run the suite of power on self-tests provided by the FIPS 140-2 cryptographic module immediately after the generation of a key.

FPT_TST_(EXP).5.3 - The TSF shall restrict the ability to invoke these self-tests to an Administrator.

167 *Application Note: For element 5.2, the Cryptographic Administrator has the ability to enable and disable this capability; this is specified in FMT_MOF.1(2). This requirement goes beyond what is required in FIPS140-2 for self-tests, in that the self-tests must be executable on demand rather than just at power-up. Conditional self-tests, such as continuous random number detection, shall not be subject to this requirement.*

### 5.1.7 Resource Utilization (FRU)

#### 5.1.7.1 Maximum quotas (transport-layer quotas) (FRU_RSA.1(1))

FRU_RSA.1.1(1) – **Refinement:** The TSF shall enforce maximum quotas of the following resources: [transport-layer representation] that a source subject identifier can use over a specified period of time.

168 *Application Note: "transport-layer representation" refers specifically to the TCP SYN attack, where half-open connections are established thus exhausting the connection table resource. The selection for this requirement was refined to specify a source subject identifier, which is more accurate than user or subject in the context of a router. If the TOE does not implement the TCP/IP protocol, this requirement would apply to a similar type of transport-layer entity for that TOE's protocol stack.*

#### 5.1.7.2 Maximum quotas (controlled connection-oriented quotas) (FRU_RSA.1(2))

FRU_RSA.1.1(2) – **Refinement:** The TSF shall enforce **administrator-specified** maximum quotas of the following resources: [controlled connection-oriented resources] that **users associated with** [*an administrator-specified network identifier and a set of administrator-specified network identifiers]* can use over an **administrator**-specified period of time.

169 *Application Note: This requirement applies to a network entity attempting to exhaust the specified connection-oriented resources (or set of such resources) on the TOE. Connectionless sessions are not a concern because they do not consume resources that persist like connection-oriented sessions do.*

170 *The ST author should fill in the first assignment with the list of connection-oriented resources to which this requirement applies. That is, when a network entity uses such a connection-oriented resource (or a collection of these resources), the TOE*

*tracks that use for the purpose of determining whether the entity has exceed the quota established by the administrator.*

171 *The ST author should use the first selection to indicate whether the TOE is able to track the assignment of the specified resources based on a single network identifier (e.g., a specific IP address) or multiple network identifiers (e.g., a specific IP subnet address). The second selection should reflect the way in which the TOE tracks such resource use. Note that the ST author may have to iterate this requirement if different resources can be controlled differently by the TOE. The ST author should ensure that FMT_MTD.2(2) specifies the actions that are taken for each resource on which there is a quota.*

### 5.1.8 TOE Access (FTA)

#### 5.1.8.1 TSF-initiated termination (FTA_SSL.3)

FTA_SSL.3.1 The TSF shall terminate an interactive session after a [assignment: time interval of user inactivity].

#### 5.1.8.2 Default TOE access banners (FTA_TAB.1)

FTA_TAB.1.1 **Refinement:** Before establishing a user session **that requires authentication**, the TSF shall display **only** a **Security Administrator specified** advisory **notice and consent** warning message regarding unauthorized use of the TOE.

172 *Application Note: The access banner applies whenever the TOE will provide a prompt for identification and authentication. The intent of this requirement is to advise users of warnings regarding the unauthorized use of the TOE and to provide the Security Administrator with control over what is displayed (e.g., if the Security Administrator chooses, they can remove banner information that informs the user of the product and version number).*

#### 5.1.8.3 TOE session establishment (FTA_TSE.1)

FTA_TSE.1.1 The TSF shall be able to deny session establishment based on [assignment: attributes].

### 5.1.9 Trusted Path/Channels (FTP)

#### 5.1.9.1 Inter-TSF trusted channel (Prevention of Disclosure) (FTP_ITC.1(1))

FTP_ITC.1.1(1) - **Refinement:** The TSF shall **use encryption to** provide a **trusted** communication channel between itself and **authorized IT entities** that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from disclosure.

FTP_ITC.1.2(1) **Refinement:** The TSF shall permit the TSF, **or the authorized IT entities** to initiate communication via the trusted channel.

173 *Application Note: The encryption used to protect the communication channel from disclosure is one of the symmetric algorithms specified in FCS_COP.1(1).*

174 *FTP_ITC.1.2 is used to ensure secure communications between the TOE and authorized IT entities (e.g., peer router). While these authorized IT entities may initiate communications, it may be the case that the TOE is required to perform a "pull" operation (e.g., obtaining routing information from a peer router).*

FTP_ITC.1.3(1) - The TSF shall initiate communication via the trusted channel for [all authentication functions, [selection: [assignment: list of other functions for which a trusted channel is required], none]].

175 *Application Note: The "other functions" are the services that are provided by the authorized IT entities (e.g., RIP).*

#### 5.1.9.2 Inter-TSF trusted channel (Detection of Modification) (FTP_ITC.1(2))

FTP_ITC.1.1(2) - **Refinement:** The TSF shall **use a cryptographic signature** to provide a **trusted** communication channel between itself and **authorized IT entities** that is logically distinct from other communication channels and provides assured identification of its end points and detection **of the modification of data**.

FTP_ITC.1.2(2) - **Refinement:** The TSF shall permit the TSF, **or the authorized IT entities** to initiate communication via the trusted channel.

176 *Application Note: The method used to provide detection of data modification transmitted through the communication channel is the cryptographic digital signature algorithm specified in FCS_COP.1(2).*

177 *FTP_ITC.1.2 is used to ensure secure communications between the TOE and authorized IT entities (e.g., peer router). While these authorized IT entities may*

*initiate communications, it may be the case that the TOE is required to perform a "pull" operation (e.g., obtaining routing information from a peer router).*

FTP_ITC.1.3(2) - The TSF shall initiate communication via the trusted channel for [all authentication functions, [selection: [assignment: list of other functions for which a trusted channel is required], none]].

178 *Application Note: The "other functions" are the services that are provided by the authorized IT entities (e.g., RIP).*

### 5.1.9.3 Trusted path (Prevention of Disclosure) (FTP_TRP.1(1) )

FTP_TRP.1.1(1) - **Refinement:** The TSF shall provide **an encrypted** communication path between itself and *remote* **administrators and authenticated users** that is logically distinct from other communication paths and provides assured identification of its end points and protection of the communicated data from disclosure.

FTP_TRP.1.2(1) - The TSF shall permit remote users to initiate communication via the trusted path.

FTP_TRP.1.3(1) – **Refinement:** The TSF shall require the use of the trusted path for user authentication, all remote administration actions, [selection: [assignment: other services for which trusted path is required, none]].

179 *Application Note: The encryption used to protect the communication channel from disclosure isone of the symmetric algorithms specified in FCS_COP.1(1)*

180 *"all remote administration actions" means that the entire remote administration session is protected with the trusted path; that is, the administrator is assured of communicating with the TOE and the data passing between the administrator and the TOE are protected from disclosure.*

### 5.1.9.4 Trusted path (Detection of Modification) (FTP_TRP.1(2))

FTP_TRP.1.1(2) - **Refinement:** The TSF shall **use a cryptographic signature to** provide a communication path between itself and remote **administrators and authenticated** users that is logically distinct from other communication paths and provides assured identification of its end points and **detection of the modification of data**.

FTP_TRP.1.2(2) - The TSF shall permit remote users to initiate communication via the trusted path.

FTP_TRP.1.3(2) – **Refinement:** The TSF shall require the use of the trusted path for user authentication, all remote administration actions, [selection: [assignment: other services for which trusted path is required, none]].

181 *Application Note: The method used to provide detection of data modification transmitted through the communication channel is the cryptographic digital signature algorithm specified in FCS_COP.1(2).*

182 *"all remote administration actions" means that the entire remote administration session is protected with the trusted path; that is, the administrator is assured of communicating with the TOE and the data passing between the administrator and the TOE provides a means for detecting the modification of data that flows through the protected communication path.*

## 5.2 Security Requirements for the IT Environment

183 This PP does not require any security requirements on the IT environment; therefore no additional rationale is needed here.

## 5.3 TOE Security Assurance Requirements

184 The TOE assurance requirements for this PP no longer map to a CC EAL in accordance with Medium Robustness for Environments Guidance dated 1 March 2004. The assurance requirements are summarized in the Table 8 below. The objectives and application notes for the explicit ADV requirements are contained in Appendix E. The methodology for performing the evaluation activities pertaining to the explicit assurance requirements is provided by CCEVS management in a separate document.

**Table 8 Assurance Requirements**

| Assurance Class | Assurance Components | |
|---|---|---|
| Configuration Management | ACM_AUT.1 | Partial CM automation |
| | ACM_CAP.4 | Generation support and acceptance procedures |
| | ADM_SCP.2 | Problem tracking CM coverage |
| Delivery and Operation | ADO_DEL.2 | Detection of modification |
| | ADO_IGS.1 | Installation, generation, and start-up procedures |

| Assurance Class | Assurance Components | |
|---|---|---|
| Development | ADV_ARC_(EXP).1 | Architectural design with justification |
| | ADV_FSP_(EXP).1 | Functional specification with complete summary |
| | ADV_HLD_(EXP).1 | Security-enforcing high-level design |
| | ADV_IMP.2 | Implementation of the TSF |
| | ADV_INT_(EXP).1 | Modular decomposition |
| | ADV_LLD_(EXP).1 | Security-enforcing low-level design |
| | ADV_RCR.1 | Informal correspondence demonstration |
| | ADV_SPM.1 | Informal TOE security policy model |
| Guidance Documents | AGD_ADM.1 | Administrator guidance |
| | AGD_USR.1 | User guidance |
| Life Cycle Support | ALC_DVS.1 | Identification of security measures |
| | ALC_FLR.2 | Flaw reporting procedures |
| | ALC_LCD.1 | Developer defined life-cycle model |
| | ALC_TAT.1 | Well-defined development tools |
| Tests | ATE_COV.2 | Analysis of coverage |
| | ATE_DPT.2 | Testing: low-level design |
| | ATE_FUN.1 | Functional testing |
| | ATE_IND.2 | Independent testing – sample |

| Assurance Class | Assurance Components | |
|---|---|---|
| Vulnerability Assessment | AVA_CCA_(EXP).2 | Systematic cryptographic module covert channel analysis |
| | AVA_MSU.2 | Validation of analysis |
| | AVA_SOF.1 | Strength of TOE security functional evaluation |
| | AVA_VLA.3 | Moderately resistant |

### 5.3.1 Configuration Management (ACM)

#### 5.3.1.1 Partial CM automation (ACM_AUT.1)

Developer action elements:

ACM_AUT.1.1D The developer shall use a CM system.

ACM_AUT.1.2D The developer shall provide a CM plan.

Content and presentation of evidence elements:

ACM_AUT.1.1C The CM system shall provide an automated means by which only authorized changes are made to the TOE implementation representation.

ACM_AUT.1.2C The CM system shall provide an automated means to support the generation of the TOE.

ACM_AUT.1.3C The CM plan shall describe the automated tools used in the CM system.

ACM_AUT.1.4C The CM plan shall describe how the automated tools are used in the CM system.

Evaluator action elements:

ACM_AUT.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

#### 5.3.1.2 Generation support and acceptance procedures (ACM_CAP.4)

Developer action elements:

ACM_CAP.4.1D The developer shall provide a reference for the TOE.

ACM_CAP.4.2D The developer shall use a CM system.

ACM_CAP.4.3D The developer shall provide CM documentation.

Content and presentation of evidence elements:

ACM_CAP.4.1C The reference for the TOE shall be unique to each version of the TOE.

ACM_CAP.4.2C The TOE shall be labeled with its reference.

ACM_CAP.4.3C The CM documentation shall include a configuration list, a CM plan, and an acceptance plan.

ACM_CAP.4.4C The configuration list shall describe the configuration items that comprise the TOE.

ACM_CAP.4.5C The CM documentation shall describe the method used to uniquely identify the configuration items.

ACM_CAP.4.6C The CM system shall uniquely identify all configuration items.

ACM_CAP.4.7C The CM plan shall describe how the CM system is used.

ACM_CAP.4.8C The evidence shall demonstrate that the CM system is operating in accordance with the CM plan.

ACM_CAP.4.9C The CM documentation shall provide evidence that all configuration items have been and are being effectively maintained under the CM system.

ACM_CAP.4.10C The CM system shall provide measures such that only authorized changes are made to the configuration items.

ACM_CAP.4.11C The CM system shall support the generation of the TOE.

ACM_CAP.4.12C The acceptance plan shall describe the procedures used to accept modified or newly created configuration items as part of the TOE.

Evaluator action elements:

ACM_CAP.4.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

#### 5.3.1.3 Problem tracking CM coverage (ACM_SCP.2)

Developer action elements:

ACM_SCP.2.1D The developer shall provide CM documentation.

Content and presentation of evidence elements:

ACM_SCP.2.1C The CM documentation shall show that the CM system, as a minimum, tracks the following: the TOE implementation representation, design documentation, test documentation, user documentation, administrator documentation, CM documentation, and security flaws.

ACM_SCP.2.2C The CM documentation shall describe how configuration items are tracked by the CM system.

Evaluator actions elements:

ACM_SCP.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### 5.3.2 Delivery and Operation (ADO)

#### 5.3.2.1 Detection of modification (ADO_DEL.2)

Developer action elements:

ADO_DEL.2.1D The developer shall document procedures for delivery of the TOE or parts of it to the user.

ADO_DEL.2.2D The developer shall use the delivery procedures.

Content and presentation of evidence elements:

ADO_DEL.2.1C The delivery documentation shall describe all procedures that are necessary to maintain security when distributing versions of the TOE to a user's site.

ADO_DEL.2.2C The delivery documentation shall describe how the various procedures and technical measures provide for the detection of modifications, or any discrepancy between the developer's master copy and the version received at the user site.

ADO_DEL.2.3C The delivery documentation shall describe how the various procedures allow detection of attempts to masquerade as the developer, even in cases in which the developer has sent nothing to the user's site.

Evaluator action elements:

ADO_DEL.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

#### 5.3.2.2 Installation, generation, and start-up procedures (ADO_IGS.1)

Developer action elements:

ADO_IGS.1.1D The developer shall document procedures necessary for the secure installation, generation, and start-up of the TOE.

Content and presentation of evidence elements:

ADO_IGS.1.1C The documentation shall describe the steps necessary for secure installation, generation, and start-up of the TOE.

Evaluator action elements:

ADO_IGS.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADO_IGS.1.2E The evaluator shall determine that the installation, generation, and start-up procedures result in a security configuration.

### 5.3.3 Development (ADV)

#### 5.3.3.1 Architectural design with justification (ADV_ARC_(EXP).1)

Developer action elements:

ADV_ARC_(EXP).1.1D The developer shall provide the architectural design of the TSF.

Content and presentation of evidence elements:

ADV_ARC_(EXP).1.1C The presentation of the architectural design of the TSF shall be informal.

ADV_ARC_(EXP).1.2C The architectural design shall be internally consistent.

ADV_ARC_(EXP).1.3C The architectural design shall describe the design of the TSF self-protection mechanisms.

ADV_ARC_(EXP).1.4C The architectural design shall describe the design of the TSF in detail sufficient to determine that the security enforcing mechanisms cannot be bypassed.

ADV_ARC_(EXP).1.5C The architectural design shall justify that the design of the TSF achieves the self-protection function.

Evaluator action elements:

ADV_ARC_(EXP).1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_ARC_(EXP).1.2E The evaluator shall analyze the architectural design and dependent documentation to determine that FPT_SEP and FPT_RVM are accurately implemented in the TSF.

#### 5.3.3.2 Functional specification with complete summary (ADV_FSP_(EXP).1)

Developer action elements:

ADV_FSP_(EXP).1.1D The developer shall provide a functional specification.

Content and presentation of evidence elements:

ADV_FSP_(EXP).1.1C The functional specification shall completely represent the TSF.

ADV_FSP_(EXP).1.2C The functional specification shall be internally consistent.

ADV_FSP_(EXP).1.3C The functional specification shall describe the external TSF interfaces (TSFIs) using an informal style.

ADV_FSP_(EXP).1.4C The functional specification shall designate each external TSFI as security enforcing or security supporting.

ADV_FSP_(EXP).1.5C The functional specification shall describe the purpose and method of use for each external TSFI.

ADV_FSP_(EXP).1.6C The functional specification shall identify and describe all parameters associated with each external TSFI.

ADV_FSP_(EXP).1.7C For security enforcing external TSFIs, the functional specification shall describe the security enforcing effects and security enforcing exceptions.

ADV_FSP_(EXP).1.8C For security enforcing external TSFIs, the functional specification shall describe direct error messages resulting from security enforcing effects and exceptions.

Evaluator action elements:

ADV_FSP_(EXP).1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_FSP_(EXP).1.2E The evaluator shall determine that the functional specification is an accurate and complete instantiation of the user-visible TOE security functional requirements.

185 *Application Note: This requirement can potentially be met by a combination of documents provided by the developer, including the Security Target and external interface specification.*

### 5.3.3.3 Security-enforcing high-level design (ADV_HLD_(EXP).1)

Developer action elements:

ADV_HLD_(EXP).1.1D The developer shall provide the high-level design of the TOE.

Content and presentation of evidence elements:

ADV_HLD_(EXP).1.1C The high-level design shall describe the structure of the TOE in terms of subsystems.

ADV_HLD_(EXP).1.2C The high-level design shall be internally consistent.

ADV_HLD_(EXP).1.3C The high level design shall describe the subsystems using an informal style.

ADV_HLD_(EXP).1.4C The high-level design shall describe the design of the TOE in sufficient detail to determine what subsystems of the TOE are part of the TSF.

ADV_HLD_(EXP).1.5C The high-level design shall identify all subsystems in the TSF, and designate them as either security enforcing or security supporting.

ADV_HLD_(EXP).1.6C The high-level design shall describe the structure of the security-enforcing subsystems.

ADV_HLD_(EXP).1.7C For security-enforcing subsystems, the high-level design shall describe the design of the security-enforcing behavior.

ADV_HLD_(EXP).1.8C For security-enforcing subsystems, the high-level design shall summarize any non-security-enforcing behavior.

ADV_HLD_(EXP).1.9C The high-level design shall summarize the behavior for security-supporting subsystems.

ADV_HLD_(EXP).1.10C The high-level design shall summarize all other interactions between subsystems of the TSF.

ADV_HLD_(EXP).1.11C The high-level design shall describe any interactions between the security-enforcing subsystems of the TSF.

Evaluator action elements:

ADV_HLD_(EXP).1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_HLD_(EXP).1.2E The evaluator shall determine that the high-level design is an accurate and complete instantiation of all user-visible TOE security functional requirements with the exception of FPT_SEP and FPT_RVM.

#### 5.3.3.4 Implementation of the TSF (ADV_IMP.2)

Developer action elements:

ADV_IMP.2.1D The developer shall provide the implementation representation for the entire TSF.

Content and presentation of evidence elements:

ADV_IMP.2.1C The implementation representation shall unambiguously define the TSF to a level of detail such that the TSF can be generated without further design decisions.

ADV_IMP.2.2C The implementation representation shall be internally consistent.

ADV_IMP.2.3C The implementation representation shall describe the relationships between all portions of the implementation.

Evaluator action elements:

ADV_IMP.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_IMP.2.2E The evaluator shall determine that the implementation representation is an accurate and complete instantiation of the TOE security functional requirements.

#### 5.3.3.5 Modular decomposition (ADV_INT_(EXP).1)

Developer action elements:

ADV_INT_(EXP).1.1D The developer shall design and implement the TSF using modular decomposition.

ADV_INT_(EXP).1.2D The developer shall use sound software engineering principles to achieve the modular decomposition of the TSF.

ADV_INT_(EXP).1.3D The developer shall design the modules such that they exhibit good internal structure and are not overly complex.

ADV_INT_(EXP).1.4D The developer shall design modules that implement the [FDP_IFC.1(1), FDP_IFC.1(2), FDP_IFF.1(1) and FDP_IFF.1(2) requirements] such that they exhibit only functional, sequential, communicational, or temporal cohesion, with limited exceptions.

ADV_INT_(EXP).1.5D The developer shall design the SFP-enforcing modules such that they exhibit only call or common coupling, with limited exceptions.

ADV_INT_(EXP).1.6D The developer shall implement TSF modules using coding standards that result in good internal structure that is not overly complex.

ADV_INT_(EXP).1.7D The developer shall provide a software architectural description.

Content and presentation of evidence elements:

ADV_INT_(EXP).1.1C The software architectural description shall identify the SFP-enforcing and non-SFP-enforcing modules.

ADV_INT_(EXP).1.2C The TSF modules shall be identical to those described by the low level design (ADV_LLD_(EXP).1.4C).

ADV_INT_(EXP).1.3C The software architectural description shall provide a justification for the designation of non-SFP-enforcing modules that interact with the SFP-enforcing module(s).

ADV_INT_(EXP).1.4C The software architectural description shall describe the process used for modular decomposition.

ADV_INT_(EXP).1.5C The software architectural description shall describe how the TSF design is a reflection of the modular decomposition process.

ADV_INT_(EXP).1.6C The software architectural description shall include the coding standards used in the development of the TSF.

ADV_INT_(EXP).1.7C The software architectural description shall provide a justification, on a per module basis, of any deviations from the coding standards.

ADV_INT_(EXP).1.8C The software architectural description shall include a coupling analysis that describes intermodule coupling for the SFP-enforcing modules.

ADV_INT_(EXP).1.9C The software architectural description shall provide a justification, on a per module basis, for any coupling or cohesion exhibited by SFP-enforcing modules, other than those permitted.

ADV_INT_(EXP).1.10C The software architectural description shall provide a justification, on a per module basis, that the SFP-enforcing modules are not overly complex.

Evaluator action elements:

ADV_INT_(EXP).1.1E The evaluator shall confirm that the information provided meets all the requirements for content and presentation of evidence.

ADV_INT_(EXP).1.2E The evaluator shall perform a cohesion analysis for the modules that substantiates the type of cohesion claimed for a subset of SFP-enforcing modules.

ADV_INT_(EXP).1.3E The evaluator shall perform a complexity analysis for a subset of TSF modules.

### 5.3.3.6 Security-enforcing low-level design (ADV_LLD_(EXP).1)

Developer action elements:

ADV_LLD_(EXP).1.1D The developer shall provide the low-level design of the TSF.

Content and presentation of evidence elements:

ADV_LLD_(EXP).1.1C The presentation of the low-level design shall be informal.

ADV_LLD_(EXP).1.2C The presentation of the low-level design shall be separate from the implementation representation.

ADV_LLD_(EXP).1.3C The low-level design shall be internally consistent.

ADV_LLD_(EXP).1.4C The low-level design shall identify and describe data that are common to more than one module, where any of the modules is a security-enforcing module.

ADV_LLD_(EXP).1.5C The low-level design shall describe the TSF in terms of modules, designating each module as either security-enforcing or security-supporting.

ADV_LLD_(EXP).1.6C The low level design shall describe each security-enforcing module in terms of its purpose, interfaces, return values from those interfaces, called interfaces to other modules, and global variables.

ADV_LLD_(EXP).1.7C For each security-enforcing module, the low level design shall provide an algorithmic description detailed enough to represent the TSF implementation.

186 *Application Note: An algorithmic description contains sufficient detail such that two different programmers would produce functionally-equivalent code, although data structures, programming methods, etc. may differ.*

ADV_LLD_(EXP).1.8C The low level design shall describe each security-supporting module in terms of its purpose and interaction with other modules.

Evaluator action elements:

ADV_LLD_(EXP).1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ADV_LLD_(EXP).1.2E The evaluator shall determine that the low-level design is an accurate and complete instantiation of all TOE security functional requirements, with the exception of FPT_SEP and FPT_RVM.

### 5.3.3.7 Informal correspondence demonstration (ADV_RCR.1)

Developer action elements:

ADV_RCR.1.1D The developer shall provide an analysis of correspondence between all adjacent pairs of TSF representations that are provided.

Content and presentation of evidence elements:

ADV_RCR.1.1C For each adjacent pair of provided TSF representations, the analysis shall demonstrate that all relevant security functionality of the more abstract TSF representation is correctly and completely refined in the less abstract TSF representation.

Evaluator action elements:

ADV_RCR.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### 5.3.3.8 Informal TOE security policy model (ADV_SPM.1)

Developer action elements:

ADV_SPM.1.1D The developer shall provide a TSP model.

ADV_SPM.1.2D The developer shall demonstrate correspondence between the functional specification and the TSP model.

Content and presentation of evidence elements:

ADV_SPM.1.1C The TSP model shall be informal.

ADV_SPM.1.2C The TSP model shall describe the rules and characteristics of all policies of the TSP that can be modeled.

ADV_SPM.1.3C The TSP model shall include a rationale that demonstrates that it is consistent and complete with respect to all policies of the TSP that can be modeled.

ADV_SPM.1.4C The demonstration of correspondence between the TSP model and the functional specification shall show that all of the security functions in the functional specification are consistent and complete with respect to the TSP model.

Evaluator action elements:

ADV_SPM.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### 5.3.4 Guidance Documents (AGD)

#### 5.3.4.1 Administrator guidance (AGD_ADM.1)

Developer action elements:

AGD_ADM.1.1D The developer shall provide administrator guidance addressed to system administrative personnel.

Content and presentation of evidence elements:

AGD_ADM.1.1C The administrator guidance shall describe the administrative functions and interfaces available to the administer of the TOE.

AGD_ADM.1.2C The administrator guidance shall describe how to administer the TOE in a secure manner.

AGD_ADM.1.3C The administrator guidance shall contain warnings about functions and privileges that should be controlled in a secure processing environment.

AGD_ADM.1.4C The administrator guidance shall describe all assumptions regarding user behavior that are relevant to secure operation of the TOE.

AGD_ADM.1.5C The administrator guidance shall describe all security parameters under the control of the administrator, indicating secure values as appropriate.

AGD_ADM.1.6C The administrator guidance shall describe each type of security-relevant event relative to the administrative functions that need to be performed, including changing the security characteristics of entities under the control of the TSF.

AGD_ADM.1.7C The administrator guidance shall be consistent with all other documentation supplied for evaluation.

AGD_ADM.1.8C The administrator guidance shall describe all security requirements for the IT environment that are relevant to the administrator.

Evaluator action elements:

AGD_ADM.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

#### 5.3.4.2 User Guidance (AGD_USR.1)

Developer action elements:

AGD_USR.1.1D The developer shall provide user guidance.

Content and presentation of evidence elements:

AGD_USR.1.1C The user guidance shall describe the functions and interfaces available to the non-administrative users of the TOE.

AGD_USR.1.2C The user guidance shall describe the use of user-accessible security functions provided by the TOE.

AGD_USR.1.3C The user guidance shall contain warnings about user-accessible functions and privileges that should be controlled in a secure processing environment.

AGD_USR.1.4C The user guidance shall clearly present all user responsibilities necessary for secure operation of the TOE, including those related to assumptions regarding user behavior found in the statement of TOE security environment.

AGD_USR.1.5C The user guidance shall be consistent with all other documentation supplied for evaluation.

AGD_USR.1.6C The user guidance shall describe all security requirements for the IT environment that are relevant to the user.

Evaluator action elements:

AGD_USR.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### 5.3.5 Life Cycle Support (ALC)

#### 5.3.5.1 Identification of security measures (ALC_DVS.1)

Developer action elements:

ALC_DVS.1.1D The developer shall produce development security documentation.

Content and presentation of evidence elements:

ALC_DVS.1.1C The development security documentation shall describe all the physical, procedural, personnel, and other security measures that are necessary to protect the confidentiality and integrity of the TOE design and implementation in its development environment.

ALC_DVS.1.2C The development security documentation shall provide evidence that these security measures are followed during the development and maintenance of the TOE.

Evaluator action elements:

ALC_DVS.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### 5.3.5.2 Flaw reporting procedures (ALC_FLR.2)

Developer action elements:

ALC_FLR.2.1D The developer shall document the flaw remediation procedures.

ALC_FLR.2.2D The developer shall establish a procedure for accepting and acting upon user reports of security flaws and requests for corrections to those flaws.

Content and presentation of evidence elements:

ALC_FLR.2.1C The flaw remediation procedures documentation shall describe the procedures used to track all reported security flaws in each release of the TOE.

ALC_FLR.2.2C The flaw remediation procedures shall require that a description of the nature and effect of each security flaw be provided, and the stats of finding a correction to that flaw.

ALC_FLR.2.3C The flaw remediation procedures shall require that corrective actions be identified for each of the security flaws.

ALC_FLR.2.4C The flaw remediation procedures documentation shall describe the methods used to provide flaw information, corrections and guidance on corrective actions to TOE users.

ALC_FLR.2.5C The procedures for processing reported security flaws shall ensure that any reported flaws are corrected and the correction issued to TOE users.

ALC_FLR.2.6C The procedures for processing reported security flaws shall provide safeguards that any corrections to these security flaws do not introduce any new flaws.

Evaluator action elements:

ALC_FLR.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### 5.3.5.3 Developer defined life-cycle model (ALC_LCD.1)

Developer action elements:

ALC_LCD.1.1D The developer shall establish a life-cycle model to be used in the development and maintenance of the TOE.

ALC_LCD.1.2D The developer shall provide life-cycle definition documentation.

Content and presentation of evidence elements:

ALC_LCD.1.1C The life-cycle definition documentation shall describe the model used to develop and maintain the TOE.

ALC_LCD.1.2C The life-cycle model shall provide for the necessary control over the development and maintenance of the TOE.

Evaluator action elements:

ALC_LCD.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

#### 5.3.5.4 Well-defined development tools (ALC_TAT.1)

Developer action elements:

ALC_TAT.1.1D The developer shall identify the development tools being used for the TOE.

ALC_TAT.1.2D The developer shall document the selected implementation-dependent options of the development tools.

Content and presentation of evidence elements:

ALC_TAT.1.1C All development tools used for implementation shall be well-defined.

ALC_TAT.1.2C The documentation of the development tools shall unambiguously define the meaning of all statements used in the implementation.

ALC_TAT.1.3C The documentation of the development tools shall unambiguously define the meaning of all implementation-dependent options.

Evaluator action elements:

ALC_TAT.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### 5.3.6 Tests (ATE)

#### 5.3.6.1 Analysis Coverage (ATE_COV.2)

Developer action elements:

ATE_COV.2.1D The developer shall provide an analysis of the test coverage.

Content and presentation of evidence elements:

ATE_COV.2.1C The analysis of the test coverage shall demonstrate the correspondence between the tests identified in the test documentation and the TSF as described in the functional specification.

ATE_COV.2.2C The analysis of the test coverage shall demonstrate that the correspondence between the TSF as described in the functional specification and the tests identified in the test documentation is complete.

Evaluator action elements:

ATE_COV.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

#### 5.3.6.2 Testing: low-level design (ATE_DPT.2)

Developer action elements:

ATE_DPT.1.1D The developer shall provide the analysis of the depth of testing.

Content and presentation of evidence elements:

ATE_DPT.1.1C The depth analysis shall demonstrate that the tests identified in the test documentation are sufficient to demonstrate that the TSF operates in accordance with its high-level design and low-level design.

Evaluator action elements:

ATE_DPT.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### 5.3.6.3 Functional testing (ATE_FUN.1)

Developer action elements:

ATE_FUN.1.1D The developer shall test the TSF and document the results.

ATE_FUN.1.2D The developer shall provide test documentation.

Content and presentation of evidence elements:

ATE_FUN.1.1C The test documentation shall consist of test plans, test procedure descriptions, expected test results and actual test results.

ATE_FUN.1.2C The test plans shall identify the security functions to be tested and describe the goal of the tests to be performed.

ATE_FUN.1.3C The test procedure descriptions shall identify the tests to be performed and describe the scenarios for testing each security function. These scenarios shall include any ordering dependencies on the results of other tests.

ATE_FUN.1.4C The expected test results shall show the anticipated outputs from a successful execution of the tests.

ATE_FUN.1.5C The test results from the developer execution of the tests shall demonstrate that each tested security function behaved as specified.

Evaluator action elements:

ATE_FUN.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

### 5.3.6.4 Independent testing – sample (ATE_IND.2)

Developer action elements:

ATE_IND.2.1D The developer shall provide the TOE for testing.

Content and presentation of evidence elements:

ATE_IND.2.1C The TOE shall be suitable for testing.

ATE_IND.2.2C The developer shall provide an equivalent set of resources to those that were used in the developer's functional testing of the TSF.

Evaluator action elements:

ATE_IND.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

ATE_IND.2.2E The evaluator shall test a subset of the TSF as appropriate to confirm that the TOE operates as specified.

ATE_IND.2.3E The evaluator shall execute a sample of tests in the test documentation to verify the developer test results.

### 5.3.7 Vulnerability Assessment (AVA)

#### 5.3.7.1 Systematic cryptographic module covert channel analysis (AVA_CCA_(EXP).2)

187 *Application Note: The covert channel analysis is performed on the entire TSF to determine that TSF interfaces cannot be used covertly to obtain critical security parameters; a search is made for the leakage of critical security parameters, rather than a violation of an information control policy.*

Developer action elements:

AVA_CCA_(EXP).2.1D The developer shall conduct a search for covert channels for the leakage of critical security parameters.

AVA_CCA_(EXP).2.2D The developer shall provide covert channel analysis documentation.

Content and presentation of evidence elements:

AVA_CCA_(EXP).2.1C The analysis documentation shall identify covert channels that leak critical security parameters and estimate their capacity.

AVA_CCA_(EXP).2.2C The analysis documentation shall describe the procedures used for determining the existence of covert channels that leak critical security parameters, and the information needed to carry out the covert channel analysis.

AVA_CCA_(EXP).2.3C The analysis documentation shall describe all assumptions made during the covert channel analysis.

AVA_CCA_(EXP).2.4C The analysis documentation shall describe the method used for estimating channel capacity, based on worst-case scenarios.

AVA_CCA_(EXP).2.5C The analysis documentation shall describe the worst-case exploitation scenario for each identified covert channel.

AVA_CCA_(EXP).2.6C The analysis documentation shall provide evidence that the method used to identify covert channels is systematic.

Evaluator action elements:

AVA_CCA_(EXP).2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA_CCA_(EXP).2.3E The evaluator shall selectively validate the covert channel analysis through independent analysis and testing.

188 *Application Note: The cryptographic security parameters are defined in FIPS 140-2.*

### 5.3.7.2 Validation of analysis (AVA_MSU.2)

Developer action elements:

AVA_MSU.2.1D The developer shall provide guidance documentation.

AVA_MSU.2.2D The developer shall document an analysis of the guidance documentation.

Content and presentation of evidence elements:

AVA_MSU.2.1C The guidance documentation shall identify all possible modes of operation of the TOE (including operation following failure or operational error), their consequences and implications for maintaining secure operation.

AVA_MSU.2.2C The guidance documentation shall be complete, clear, consistent and reasonable.

AVA_MSU.2.3C The guidance documentation shall list all assumptions about the intended environment.

AVA_MSU.2.4C The guidance documentation shall list all requirements for external security measures (including external procedural, physical and personnel controls).

AVA_MSU.2.5C The analysis documentation shall demonstrate that the guidance documentation is complete.

Evaluator action elements:

AVA_MSU.2.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA_MSU.2.2E The evaluator shall repeat all configuration and installation procedures, and other procedures selectively, to confirm that the TOE can be configured and used securely using only the supplied guidance documentation.

AVA_MSU.2.3E The evaluator shall determine that the use of the guidance documentation allows all insecure states to be detected.

AVA_MSU.2.4E The evaluator shall confirm that the analysis documentation shows that guidance is provided for secure operation in all modes of operation of the TOE.

### 5.3.7.3 Strength of TOE security function evaluation (AVA_SOF.1)

Developer action elements:

AVA_SOF.1.1D The developer shall perform a strength of TOE security function analysis for each mechanism identified in the ST as having a strength of TOE security function claim.

Content and presentation of evidence elements:

AVA_SOF.1.1C For each mechanism with a strength of TOE security function claim, the strength of TOE security function analysis shall show that it meets or exceeds the minimum strength level of SOF-basic.

AVA_SOF.1.2C For each mechanism with a specific strength of TOE security function claim the strength of TOE security function analysis shall show that it meets or exceeds the specific strength of function metric of SOF-basic.

Evaluator action elements:

AVA_SOF.1.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA_SOF.1.2E The evaluator shall confirm that the strength claims are correct.

### 5.3.7.4 Moderately resistant (AVA_VLA.3)

Developer action elements:

AVA_VLA.3.1D The developer shall perform and document an analysis of the TOE deliverables searching for ways in which a user can violate the TSP.

AVA_VLA.3.2D The developer shall document the disposition of identified vulnerabilities.

Content and presentation of evidence elements:

AVA_VLA.3.1C The documentation shall show, for all identified vulnerabilities, that the vulnerability cannot be exploited in the intended environment for the TOE.

AVA_VLA.3.2C The documentation shall justify that the TOE, with the identified vulnerabilities, is resistant to obvious penetration attacks.

AVA_VLA.3.3C The evidence shall show that the search for vulnerabilities is systematic.

Evaluator action elements:

AVA_VLA.3.1E The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence.

AVA_VLA.3.2E The evaluator shall conduct penetration testing, building on the developer vulnerability analysis, to ensure the identified vulnerabilities have been addressed.

AVA_VLA.3.3E The evaluator shall perform an independent vulnerability analysis.

AVA_VLA.3.4E The evaluator shall perform independent penetration testing, based on the independent vulnerability analysis, to determine the exploitability of additional identified vulnerabilities in the intended environment.

AVA_VLA.3.5E The evaluator shall determine that the TOE is resistant to penetration attacks performed by an attacker possessing a moderate attack potential.

# 6 RATIONALE

189 This section provides the rationale for the selection of the IT security requirements, objectives, assumptions, and threats. In particular, it shows that the IT security requirements are suitable to meet the security objectives, which in turn are shown to be suitable to cover all aspects of the TOE security environment.

## 6.1 Rationale for TOE Security Objectives

**Table 9 Rationale for TOE Security Objectives**

| Threat/Policy | Objectives Addressing the Threat/Policy | Rationale |
|---|---|---|
| T.ADMIN_ERROR<br>An administrator may incorrectly install or configure the TOE, or install a corrupted TOE resulting in ineffective security mechanisms. | O.ROBUST_ADMIN_GUIDANCE<br>The TOE will provide administrators with the necessary information for secure delivery and management. | O.ROBUST_ADMIN_GUIDANCE (ADO_DEL.2, ADO_IGS.1, AGD_ADM.1, AGD_USR.1, AVA_MSU.2) help to mitigate this threat by ensuring the TOE administrators have guidance that instructs them how to administer the TOE in a secure manner and to provide the administrator with instructions to ensure the TOE was not corrupted during the delivery process. Having this guidance helps to reduce the mistakes that an administrator might make that could cause the TOE to be configured in a way that is insecure. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| | O.ADMIN_ROLE<br>The TOE will provide administrator roles to isolate administrative actions, and to make the administrative functions available locally and remotely. | O.ADMIN_ROLE (FMT_SMR.2) plays a role in mitigating this threat by limiting the functions an administrator can perform in a given role. |
| | O.MANAGE<br>The TOE will provide all the functions and facilities necessary to support the administrators in their management of the security of the TOE, and restrict these functions and facilities from unauthorized use. | O.MANAGE (FMT_MTD.1(1), FMT_MTD.1(2), FMT_MTD.1(3), FMT_MTD.1(4)) also contributes to mitigating this threat by providing administrators the capability to view configuration settings. For example, if the Security Administrator made a mistake when configuring the rule-set, providing them the capability to view the rules affords them the ability to review the rules and discover any mistakes that might have been made. |
| T.ADMIN_ROGUE<br>An administrator's intentions may become malicious resulting in user or TSF data being compromised. | O.ADMIN_ROLE<br>The TOE will provide administrator roles to isolate administrative actions, and to make the administrative functions available locally and remotely. | O.ADMIN_ROLE (FMT_SMR.2) mitigates this threat by restricting the functions available to an administrator. This is somewhat different than the part this objective plays in countering T.ADMIN_ERROR, in that this presumes that separate individuals will be assigned separate roles. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| T.AUDIT_COMPROMISE<br>A malicious user or process may view audit records, cause audit records to be lost or modified, or prevent future audit records from being recorded, thus masking a user's action. | O.AUDIT_PROTECTION<br>The TOE will provide the capability to protect audit information. | O.AUDIT_PROTECTION (FAU.SAR.2, FAU_STG.1-NIAP-0429, FAU_STG.3, FAU_STG.NIAP-0414-1, FMT_SMF.1) contributes to mitigating this threat by controlling access to the audit trail. The auditor and any trusted IT entities performing IDS-like functions are the only ones allowed to read the audit trail. No one is allowed to modify audit records, and the Auditor is the only one allowed to delete audit records in the audit trail. The TOE has the capability to prevent auditable actions from occurring if the audit trail is full, and of notifying an administrator if the audit trail is approaching its capacity. In addition, the TOE has the capability to restore audit data corrupted by the attacker. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| | O.RESIDUAL_INFORMATION<br>The TOE will ensure that any information contained in a protected resource is not released when the resource is reallocated. | O.RESIDUAL_INFORMATION (FDP.RIP.2) prevents a user not authorized to read the audit trail from access to audit information that might otherwise be persistent in a TOE resource (e.g., memory). By ensuring the TOE prevents residual information in a resource, audit information will not become available to any user or process except those explicitly authorized for that data. |
| | O.SELF_PROTECTION<br>The TSF will maintain a domain for its own execution that protects itself and its resources from external interference, tampering or unauthorized disclosure. | O.SELF_PROTECTION (FPT_SEP.2, FPT_RVM.1) contributes to countering this threat by ensuring that the TSF can protect itself from users. If the TSF could not maintain and control its domain of execution, it could not be trusted to control access to the resources under its control, which includes the audit trail. Likewise, ensuring that the functions that protect the audit trail are always invoked is also critical to the mitigation of this threat. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| T.CRYPTO_COMPROMISE<br>A malicious user or process may cause key, data or executable code associated with the cryptographic functionality to be inappropriately accessed (viewed, modified, or deleted), thus compromising the cryptographic mechanisms and the data protected by those mechanisms. | O.RESIDUAL_INFORMATION<br>The TOE will ensure that any information contained in a protected resource is not released when the resource is reallocated. | O.RESIDUAL_INFORMATION (FDP_RIP.2) is necessary to mitigate this threat by ensuring no TSF data remain in resources allocated to a user. Even if the security mechanisms do not allow a user to explicitly view TSF data, if TSF data were to inappropriately reside in a resource that was made available to a user, that user would be able to inappropriately view the TSF data. |
| | O.SELF_PROTECTION<br>The TSF will maintain a domain for its own execution that protects itself and its resources from external interference, tampering, or unauthorized disclosure. | O.SELF_PROTECTION (FPT_SEP.2, FPT_RVM.1) contributes to countering this threat by ensuring that the TSF can protect itself from users. If the TSF could not maintain and control its domain of execution, it could not be trusted to control access to the resources under its control, which includes the cryptographic data and executable code. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| | O.DOCUMENT_KEY_LEAKAGE<br>The bandwidth of channels that can be used to compromise key material shall be documented. | O.DOCUMENT_KEY_LEAKAGE (AVA_CCA_(EXP).2) addresses this threat by requiring the developer to perform an analysis that documents the amount of key information that can be leaked via a covert channel. This provides information that identifies how much material could be inappropriately obtained within a specified time period. |
| T.FLAWED_DESIGN | O.CHANGE_MANAGEMENT<br>The configuration of, and all changes to, the TOE and its development evidence will be analyzed, tracked, and controlled throughout the TOE's development. | O.CHANGE_MANAGEMENT (ACM_AUT.1, ACM_CAP.4, ACM_SCP.2, ALC_DVS.1, ALC_FLR.2, ALC_LCD.1) plays a role in countering this threat by requiring the developer to provide control of the changes made to the TOE's design. This includes controlling physical access to the TOE's development area, and having an automated configuration management system that ensures changes made to the TOE go through an approval process and only those persons that are authorized can make changes to the TOE's design and its documentation. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| | O.SOUND_DESIGN<br>The TOE will be designed using sound design principles and techniques. The TOE design, design principles and design techniques will be adequately and accurately documented. | O.SOUND_DESIGN (ADV_FSP_(EXP).1, ADV_HLD_(EXP).1, ADV_INT_(EXP).1, ADV_LLD_(EXP).1, ADV_ARC_(EXP).1, ADV_RCR.1, ADV_SPM.1) counters this threat, to a degree, by requiring that the TOE be developed using sound engineering principles. By accurately and completely documenting the design of the security mechanisms in the TOE, including a security model, the design of the TOE can be better understood, which increases the chances that design errors will be discovered. |
| | O.VULNERABILITY_ANALYSIS_TEST<br>The TOE will undergo appropriate independent vulnerability analysis and penetration testing to demonstrate the design and implementation of the TOE does not allow attackers with medium attack potential to violate the TOE's security policies. | O.VULNERABILITY_ANALYSIS_TEST (AVA_VLA.3) ensures that the design of the TOE is independently analyzed for design flaws. Having an independent party perform the assessment ensures an objective approach is taken and may find errors in the design that would be left undiscovered by developers that have a preconceived incorrect understanding of the TOE's design. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| T.FLAWED_IMPLEMENTATION<br>Unintentional or intentional errors in requirements specification or design of the TOE may occur, leading to flaws that may be exploited by a malicious user or program. | O.CHANGE_MANAGEMENT<br>The configuration of, and all changes to, the TOE and its development evidence will be analyzed, tracked, and controlled throughout the TOE's development. | O.CHANGE_MANAGEMENT (ACM_CAP.4, ACM_SCP.2, ALC_DVS.1, ALC_FLR.2, ALC_LCD.1, ACM_AUT.1) This objective plays a role in mitigating this threat in the same way that the flawed design threat is mitigated. By controlling who has access to the TOE's implementation representation and ensuring that changes to the implementation are analyzed and made in a controlled manner, the threat of intentional or unintentional errors being introduced into the implementation are reduced. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| | O.SOUND_IMPLEMENTATION<br>The implementation of the TOE will be an accurate instantiation of its design, and is adequately and accurately documented. | In addition to documenting the design so that implementers have a thorough understanding of the design, O.SOUND_IMPLEMENTATION (ADV_IMP.2, ADV_LLD_(EXP).1, ADV_RCR.1, ADV_INT_(EXP).1, ADV_ARC_(EXP).1, ALC_TAT.1) requires that the developer's tools and techniques for implementing the design are documented. Having accurate and complete documentation, and having the appropriate tools and procedures in the development process helps reduce the likelihood of unintentional errors being introduced into the implementation. |
| | O.THOROUGH_FUNCTIONAL_TESTING<br>The TOE will undergo appropriate security functional testing that demonstrates the TSF satisfies the security functional requirements. | Although the previous three objectives help minimize the introduction of errors into the implementation, O.THOROUGH_FUNCTIONAL_TESTING (ATE_COV.2, ATE_FUN.1, ATE_DPT.2, ATE_IND.2) increases the likelihood that any errors that do exist in the implementation (with respect to the functional specification, high level, and low-level design) will be discovered through testing. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| | O.VULNERABILITY_ANALYSIS_TEST<br>The TOE will undergo appropriate independent vulnerability analysis and penetration testing to demonstrate the design and implementation of the TOE does not allow attackers with medium attack potential to violate the TOE's security policies. | O.VULNERABILITY_ANALYSIS_TEST (AVA_VLA.3) helps reduce errors in the implementation that may not be discovered during functional testing. Ambiguous design documentation, and the fact that exhaustive testing of the external interfaces is not required may leave bugs in the implementation undiscovered in functional testing. Having an independent party perform a vulnerability analysis and conduct testing outside the scope of functional testing increases the likelihood of finding errors. |
| T.MALICIOUS_TSF_COMPROMISE<br>A malicious user or process may cause TSF data or executable code to be inappropriately accessed (viewed, modified, or deleted). | O.RESIDUAL_INFORMATION<br>The TOE will ensure that any information contained in a protected resource is not released when the resource is reallocated. | O.RESIDUAL_INFORMATION (FDP_RIP.2, FCS_CKM.4) is necessary to mitigate this threat by ensuring no TSF data remain in resources allocated to a user. Even if the security mechanisms do not allow a user to explicitly view TSF data, if TSF data were to inappropriately reside in a resource that was made available to a user, that user would be able to inappropriately view the TSF data. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| | O.SELF_PROTECTION<br>The TSF will maintain a domain for its own execution that protects itself and its resources from external interference, tampering or unauthorized disclosure. | O.SELF_PROTECTION (FPT_SEP.2, FPT_RVM.1) requires that the TSF be able to protect itself from tampering and that the security mechanisms in the TSF cannot be bypassed. Without this objective, there could be no assurance that users could not view or modify TSF data or TSF executables. |
| | O.MANAGE<br>The TOE will provide all the functions and facilities necessary to support the administrators in their management of the security of the TOE, and restrict these functions and facilities from unauthorized use. | O.MANAGE (FMT_MTD.1(1)FMT_MTD.1(2), FMT_MTD.1(3), FMT_MTD.1(4), FMT_MSA.1(1), FMT_MSA.1(2), FMT_MSA.3(1), FMT_MSA.3(2), FMT_MOF.1(1), FMT_MOF.1(2), FMT_MOF.1(3), FMT_MOF.1(4), FMT_MOF.1(5), FMT_MOF.1(6), FMT_MTD.2(1), FMT_MTD.2(2), FMT_SMF.1) provides the capability to restrict access to TSF to those that are authorized to use the functions. Satisfaction of this objective (and its associated requirements) prevents unauthorized access to TSF functions and data through the administrative mechanisms. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| | O.DISPLAY_BANNER<br>The TOE will display an advisory warning regarding use of the TOE. | O.DISPLAY_BANNER (FTA_TAB.1) helps mitigate this threat by providing the Administrator the ability to remove product information (e.g., product name, version number) from a banner that is displayed to users. Having product information about the TOE provides an attacker with information that may increase their ability to compromise the TOE. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| | O.TRUSTED_PATH<br>The TOE will provide a means to ensure that users are not communicating with some other entity pretending to be the TOE when supplying identification and authentication data. | O.TRUSTED_PATH (FTP_TRP.1(1), FTP_TRP.1(2), FTP_ITC.1(1), FTP_ITC.1(2)) plays a role in addressing this threat by ensuring that there is a trusted communication path between the TSF and various users (remote administrators, and trusted IT entities (for performing replication, for instance)). This ensures the transmitted data cannot be compromised or disclosed during the duration of the trusted path. The protection offered by this objective is limited to TSF data, including authentication data and all data sent or received by trusted IT entities (a relying party's user data is not protected; only the authentication portion of the session is protected). |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| T.MASQUERADE<br>A malicious user, process, or external IT entity may masquerade as an authorized entity in order to gain access to data or TOE resources. | O.ROBUST_TOE_ACCESS<br>The TOE will provide mechanisms that control a user's logical access to the TOE and to explicitly deny access to specific users when appropriate. | O.ROBUST_TOE_ACCESS (FIA_AFL.1, FIA_ATD.1(1), FIA_ATD.1(2), FIA_UID.2, FIA_UAU.2, FIA_UAU_(EXP).5, FTA_TSE.1, AVA_SOF.1) mitigates this threat by controlling the logical access to the TOE and its resources. By constraining how and when authorized users can access the TOE, and by mandating the type and strength of the authentication mechanisms, this objective helps mitigate the possibility of a user attempting to login and masquerade as an authorized user. In addition, this objective provides the administrator the means to control the number of failed login attempts a user can generate before an account is locked out, further reducing the possibility of a user gaining unauthorized access to the TOE. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| T.POOR_TEST<br>Lack of or insufficient tests to demonstrate that all TOE security functions operate correctly (including in a fielded TOE) may result in incorrect TOE behavior being undiscovered thereby causing potential security vulnerabilities. | O.CORRECT_ TSF_OPERATION<br>The TOE will provide a capability to test the TSF to ensure the correct operation of the TSF in its operational environment. | While these testing activities are necessary for successful completion of an evaluation, this testing activity does not address the concern that the TOE continues to operate correctly and enforce its security policies once it has been fielded. Some level of testing must be available to end users to ensure the TOE's security mechanisms continue to operate correctly once the TOE is fielded. O.CORRECT_ TSF_OPERATION (FPT_TST_(EXP).4, FPT_TST_(EXP).5) ensures that once the TOE is installed at a customer's location, the capability exists that the integrity of the TSF (hardware and software, including the cryptographic functions) can be demonstrated, and thus providing end users the confidence that the TOE's security policies continue to be enforced. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| | O.THOROUGH_FUNCTIONAL_TESTING<br>The TOE will undergo appropriate security functional testing that demonstrates the TSF satisfies the security functional requirements. | Design analysis determines that TOE's documented design satisfies the security functional requirements. In order to ensure the TOE's design is correctly realized in its implementation, the appropriate level of functional testing of the TOE's security mechanisms must be performed during the evaluation of the TOE. O.THOROUGH_FUNCTIONAL_ TESTING (ATE_FUN.1, ATE_COV.2, ATE_DPT.2, ATE_IND.2) ensures that adequate functional testing is performed to demonstrate the TSF satisfies the security functional requirements and that the TOE's security mechanisms operate as documented. While functional testing serves an important purpose, it does not ensure the TSFI cannot be used in unintended ways to circumvent the TOE's security policies. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| | O.VULNERABILITY_ANALYSIS_TEST<br>The TOE will undergo appropriate independent vulnerability analysis and penetration testing to demonstrate the design and implementation of the TOE does not allow attackers with medium attack potential to violate the TOE's security policies. | O.VULNERABILITY_ANALYSIS_TEST (AVA_VLA.3) addresses this concern by requiring a vulnerability analysis be performed in conjunction with testing that goes beyond functional testing. This objective provides a measure of confidence that the TOE does not contain security flaws that may not be identified through functional testing. |
| T.REPLAY<br>A user may gain inappropriate access to the TOE by replaying authentication information, or may cause the TOE to be inappropriately configured by replaying TSF data or security attributes (e.g., captured as transmitted during the course of legitimate use). | O.REPLAY_DETECTION<br>The TOE will provide a means to detect and reject the replay of authentication data and other TSF data and security attributes. | O.REPLAY_DETECTION (FPT_RPL.1) prevents a user from replaying authentication data. Prevention of replay of authentication data will counter the threat that a user will be able to record an authentication session between a trusted entity (administrative user or trusted IT entity) and then replay it to gain access to the TOE, and counter the ability of a user to act as another user. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| T.RESIDUAL_DATA<br>A user or process may gain unauthorized access to data through reallocation of TOE resources from one user or process to another. | O.RESIDUAL_INFORMATION<br>The TOE will ensure that any information contained in a protected resource is not released when the resource is reallocated. | O.RESIDUAL_INFORMATION (FDP_RIP.2, FCS_CKM.4) counters this threat by ensuring that TSF data and user data is not persistent when resources are released by one user/process and allocated to another user/process. This means that network packets sent in response to a request will not have residual data from another packet (potentially from another user) due to the padding of a packet. The TSF data will be zeroized once the resources are released by a user/process. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| T.RESOURCE_EXHAUSTION<br>A malicious process or user may block others from system resources (e.g., connection state tables, TCP connections) via a resource exhaustion denial of service attack. | O.RESOURCE_SHARING<br>The TOE shall provide mechanisms that mitigate attempts to exhaust connection-oriented resources provided by the TOE (e.g., entries in a connection state table; Transmission Control Protocol (TCP) connections to the TOE). | O.RESOURCE_SHARING (FRU_RSA.1(1), FRU_RSA.1(2), FMT_MTD.2(1), FMT_MTD.2(2)) mitigates this threat by requiring the TOE to provide controls relating to two different resources: CPU time and available network connections. The administrator is allowed to specify a percentage of processor time that is allowed to be used so that an attempt to exhaust the resource will fail when it reaches the quota. This objective also addresses the denial-of-service attack of a user attempting to exhaust the connection-oriented resources by generating a large number of half-open connections (e.g., SYN attack). |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| T.SPOOFING<br>A malicious user, process, or external IT entity may misrepresent itself as the TOE to obtain identification and authentication data. | O.TRUSTED_PATH<br>The TOE will provide a means to ensure that users are not communicating with some other entity pretending to be the TOE when supplying identification and authentication data. | It is possible for an entity other than the TOE (a subject on the TOE, or another IT entity on the network between the TOE and the end user) to provide an environment that may lead a user to mistakenly believe they are interacting with the TOE, thereby fooling the user into divulging identification and authentication information. O.TRUSTED_PATH (FTP_ITC.1(1), FTP_ITC.1(2) FTP_TRP.1(1), FTP_TRP.1(2)) mitigates this threat by ensuring users have the capability to ensure they are communicating with the TOE when providing identification and authentication data to the TOE. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| T.TRAFFIC_ANALYSIS<br>An attacker Collects source and destination addresses, volume of data, and time of day that messages are sent. | O.CRYPTOGRAPHIC_FUNCTIONS<br>The TOE shall provide cryptographic functions (i.e., encryption/decryption and digital signature operations) to maintain the confidentiality and allow for detection of modification of TSF data that is transmitted between physically separated portions of the TOE, or stored outside the TOE. | O.CRYPTOGRAPHIC_FUNCTIONS (FCS_CKM.1(1), FCS_CKM.1(2), FCS_CKM.2, FCS_CKM.4, , FCS_COP.1(1), FCS_COP.1(2)) mitigates this threat by providing for the use of cryptographic functions to detect when information has been modified. |
| | O.PROTECT_IN_TRANSIT<br>The TSF shall protect TSF data when it is in transit between the TSF and another trusted IT entity. | O.PROTECT_IN_TRANSIT (FPT_ITA.1, FPT_ITC.1, FPT_ITI.1, FTP_TRP.1(1), FTP_TRP.1(2), FTP_ITC.1(1), FTP_ITC.1(2) satifies this threat by ensuring protection of the communication between the TOE and trusted IT entities while transmitting data.) |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| T.UNATTENDED_SESSION<br>A user may gain unauthorized access to an unattended session. | O.ROBUST_TOE_ACCESS<br>The TOE will provide mechanisms that control a user's logical access to the TOE and to explicitly deny access to specific users when appropriate. | O. ROBUST_TOE_ACCESS (FTA_SSL.3) helps to mitigate this threat by including mechanisms that place controls on user's sessions. Local and remote administrator's sessions are dropped after an Administrator-defined time period of inactivity. Dropping the connection of a local and remote session (after the specified time period) reduces the risk of someone accessing the local and remote machines where the session was established, thus gaining unauthorized access to the session. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| T.UNAUTHORIZED_ACCESS<br>A user may gain access to user data for which they are not authorized according to the TOE security policy. | O.MEDIATE_INFORMATION_FLOW<br>The TOE must mediate the flow of information between sets of TOE network interfaces or between a network interface and the TOE itself in accordance with its security policy. | O.MEDIATE_INFORMATION_FLOW (FDP_IFC.1(1), FDP_IFC.1(2), FDP_IFF.1(1), FDP_IFF.1(2), works to mitigate this threat by ensuring that all network packets that flow through the TOE are subject to the information flow policies. One of the rules ensures that the network identifiers in a packet is in the set of network identifiers associated with a TOE's network interface. Therefore, if a user supplied a network identifier in a packet that purported to originate from a network associated with a TOE network interface other than the one the user supplied the packet on, the packet would not be allowed to flow through the TOE or access TOE services. The authenticated TOE policy ensures that user data being sent between PEER TOEs is encrypted if there is a rule (specified by the Security Administrator) that states |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| | | data is to be encrypted between those two hosts.<br><br>The authenticated TOE policy allows the administrator to specifiy each originating host (identified by IP address), which destination addresses must be access through a router and which destination addresses may be accessed without encryption. If a potential security violation has been detected, the TOE displays a message that identifies the potential security violation to all administrator consoles. The consoles include the local TOE console and any active remote administrative sessions. If an administrator is not currently accessing the TOE, the message is stored and immediately displayed the next time an administrator accesses the TOE.<br><br>The TOE restricts the ability to modify the security attributes associated with access control rules, access to authenticated and unauthenticated services, etc. to the Security Administrator. This feature |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| | | ensures that no other user can modify the information flow policy to bypass the intended TOE security policy. |
| | O.USER_GUIDANCE<br>The TOE will provide users with the information necessary to correctly use the security mechanisms. | O.USER_GUIDANCE (AGD_USR.1) mitigates this threat by providing the user the information necessary to use the security mechanisms that control access to user data in a secure manner. For instance, the method by which the discretionary access control mechanism (FDP_ACC.1, FDP_ACF.1) is configured, and how to apply it to the data the user owns, is described in the user guidance. If this information were not available to the user, the information may be left unprotected, or the user may mis-configure the controls and unintentionally allow unauthorized access to their data. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| T.UNIDENTIFIED_ACTIONS<br>The administrator may fail to notice potential security violations, thus limiting the administrator's ability to identify and take action against a possible security breach. | O.AUDIT_REVIEW<br>The TOE will provide the capability to selectively view audit information, and alert the administrator of identified potential security violations. | O.AUDIT_REVIEW (FAU_SAA.1-NIAP-0407, FAU_ARP.1, FAU_ARP_ACK_(EXP).1, FAU_ SAR.1, FAU_SAR.3) helps to mitigate this threat by providing a variety of mechanisms for monitoring the use of the system. The two basic ways audit review is performed is through analysis of the audit trail produced by the audit mechanism, and through the use of an automated analysis and alarm system.<br>For analyzing the audit trail, the TOE requires an Auditor role. This role is restricted to Audit record review and the deletion of the audit trail for maintenance purposes. A search and sort capability provides an efficient mechanism for the Audit Administrator to view pertinent audit information.<br>In addition to the local Auditor role, the TOE also |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| | | has the capability to export the audit information to an external audit analysis tool (such as an intrusion detection system) for more detailed or composite audit analysis.<br><br>The TOE's audit analysis mechanism must consist of a minimum set of configurable audit events that could indicate a potential security violation. Thresholds for these events must be configurable by an appropriate administrative role. By configuring these auditable events, the TOE monitors the occurrences of these events (e.g. set number of authentication failures, set number directory access failures, self-test failures, etc.) and immediately notifies an administrator once an event has occurred or a set threshold has been met. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| T.UNAUTHORIZED_PEER<br>An unauthorized IT entity may attempt to establish a security association with the TOE. | O.PEER_AUTHENTICATION<br>The TOE will authenticate each peer TOE that attempts to establish a security association with the TOE. | O.PEER_AUTHENTICATION (FCS_IKE_(EXP).1) mitigates this threat by requiring that the TOE implement the Internet Key Exchange protocol, as specified in RFC2409, to establish a secure, authenticated channel between the TOE and another remote router before establishing a security association with that router. |
| T.UNKNOWN_STATE<br>When the TOE is initially started or restarted after a failure, the security state of the TOE may be unknown. | O.MAINT_MODE<br>The TOE shall provide a mode from which recovery or initial startup procedures can be performed. | O.MAINT_MODE (FPT_RCV.2) helps to mitigate this threat by ensuring that the TOE does not continue to operate in an insecure state when a hardware or software failure occurs. After a failure, the TOE enters a state that disallows operations and requires an administrator to follow documented procedures to return the TOE to a secure state. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| | O.CORRECT_ TSF_OPERATION<br>The TOE will provide a capability to test the TSF to ensure the correct operation of the TSF in its operational environment. | O.CORRECT_TSF_OPERATION (FPT_TST_(EXP).4, FPT_TST. _(EXP).5) counters this threat by ensuring that the TSF runs a suite of tests to successfully demonstrate the correct operation of the TSF (hardware and software) and the TSF's cryptographic components at initial startup of the TOE. In addition to ensuring that the TOE's security state can be verified, an administrator can verify the integrity of the TSF's data and stored code and the TSF's cryptographic data and stored code using the TOE-provided cryptographic mechanisms. |
| | O.SOUND_DESIGN<br>The TOE will be designed using sound design principles and techniques. The TOE design, design principles and design techniques will be adequately and accurately documented. | O.SOUND_DESIGN (ADV_SPM.1) works to mitigate this threat by requiring that the TOE developers provide accurate and complete design documentation of the security mechanisms in the TOE, including a security model. By providing this documentation, the possible secure states of the TOE are described, thus enabling the administrator to return the TOE to one of these states during the recovery process. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| | O.ROBUST_ADMIN_GUIDANCE<br>The TOE will provide administrators with the necessary information for secure delivery and management. | O. ROBUST_ADMIN_GUIDANCE (ADO_IGS.1, AGD_ADM.1) provides administrative guidance for the secure start-up of the TOE and guidance to configure and administer the TOE securely. This guidance provides administrators with the information necessary to ensure that the TOE is started and initialized in a secure manor. The guidance also provides information about the corrective measure necessary when a failure occurs (i.e., how to bring the TOE back into a secure state). |
| P.ACCESS_BANNER<br>The TOE shall display an initial banner describing restrictions of use, legal agreements, or any other appropriate information to which users consent by accessing the TOE. | O.DISPLAY_BANNER<br>The TOE will display an advisory warning regarding use of the TOE. | O.DISPLAY_BANNER (FTA_TAB.1) satisfies this policy by ensuring that the TOE displays an Administrator-configurable banner that provides all users with a warning about the unauthorized use of the TOE. This is required to be displayed before an interactive administrative session. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| P.ACCOUNTABILITY<br>The authorized users of the TOE shall be held accountable for their actions within the TOE. | O.AUDIT_GENERATION<br>The TOE will provide the capability to detect and create records of security-relevant events associated with users. | O.AUDIT_GENERATION (FAU_GEN.1-NIAP-0407, FAU_GEN.2-NIAP-410, FIA_USB.1, FAU_SEL.1-NIAP-0407) addresses this policy by providing an audit mechanism to record the actions of a specific user, and the capability for an administrator to "pre-select" audit events based on the user ID. The audit event selection function is configurable during run-time to ensure the TOE is able to capture security-relevant events given changes in threat conditions. Additionally, the administrator's ID is recorded when any security relevant change is made to the TOE (e.g., access rule modification, start-stop of the audit mechanism, establishment of a trusted channel, etc.). Attributes used in the audit record generation process are also required to be bound to the subject, ensuring users are held accountable |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| | O.TIME_STAMPS<br>The TOE shall provide reliable time stamps and the capability for the administrator to set the time used for these time stamps. | O.TIME_STAMPS (FPT_STM.1, FMT_MTD.1) plays a role in supporting this policy by requiring the TOE to provide a reliable time stamp. The audit mechanism is required to include the current date and time in each audit record. All audit records that include the user ID will also include the date and time that the event occurred. |
| | O.ROBUST_TOE_ACCESS<br>The TOE will provide mechanisms that control a user's logical access to the TOE and to explicitly deny access to specific users when appropriate. | O.ROBUST_TOE_ACCESS (FIA_UID.2, FIA_UAU.2, FIA_UAU_(EXP).5) supports this policy by requiring the TOE to identify and authenticate all authorized users prior to allowing any TOE access or any TOE mediated access on behalf of those users. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| P.ADMIN_ACCESS<br>Administrators shall be able to administer the TOE both locally and remotely through protected communications channels. | O.ADMIN_ROLE<br>The TOE will provide administrator roles to isolate administrative actions, and to make the administrative functions available locally and remotely. | O.ADMIN_ROLE (FMT_SMR.2) supports this policy by requiring the TOE to provide mechanisms (e.g., local authentication, remote authentication, means to configure and manage the TOE both remotely and locally) that allow remote and local administration of the TOE. This is not to say that everything that can be done by a local administrator must also be provided to the remote administrator. In fact, it may be desirable to have some functionality restricted to the local administrator. |
| | O.TRUSTED_PATH<br>The TOE will provide a means to ensure that users are not communicating with some other entity pretending to be the TOE when supplying identification and authentication data. | O.TRUSTED_PATH (FTP_TRP.1(1), FTP_TRP.1(2), FTP_ITC.1(1), FTP_ITC.1(2)) satisfies this policy by requiring that each remote administrative and management session for all trusted users is authenticated and conducted via a secure channel. Additionally, all trusted IT entities (e.g., trusted peer directories, intrusion detection systems) connect through a protected channel, thus avoiding disclosure and spoofing problems. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| P.COMPATIBILITY<br>The TOE must meet RFC requirements for implemented protocols to facilitate inter-operation with other routers and network equipment using the same protocols. | O.PROTOCOLS<br>The TOE will ensure that standardized protocols are implemented in the TOE to RFC and/or Industry specifications to ensure interoperability. | O.PROTOCOLS (FPT_FLS.1, FPT_PRO_(EXP).1) satifies this policy by requiring that standardized protocols are implemented in the TOE to ensure interoperatibility among peer TOEs therefore not compromising the secure state of the router. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| P.CRYPTOGRAPHIC_FUNCTIONS<br>The TOE shall provide cryptographic functions for its own use, including encryption/decryption and digital signature operations. | O.CRYPTOGRAPHIC_FUNCTIONS<br>The TOE shall provide cryptographic functions (i.e., encryption/decryption and digital signature operations) to maintain the confidentiality and allow for detection of modification of TSF data that is transmitted between physically separated portions of the TOE, or stored outside the TOE. | O.CRYPTOGRAPHIC_FUNCTIONS (FCS_CKM.1(1), FCS_CKM.1(2), FCS_CKM.2, FCS_CKM.4, FCS_CKM_(EXP).1, FCS_CKM_(EXP).2, FCS_COA_(EXP).1, FCS_COP.1(1), FCS_COP.1(2), FCS_COP.1(3), FCS_COP.1(4), FCS_COP.1(5)) FCS_COP_(EXP).1 implements this policy, requiring a combination of FIPS-validation and non-FIPS-validated cryptographic mechanisms that are used to provide encryption/decryption services, and digital signature functions. Functions include symmetric encryption and decryption, digital signatures, and key generation and establishment functions. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| P.CRYPTOGRAPHY_VALIDATED<br>Where the TOE requires FIPS-approved security functions, only NIST FIPS Publication validated cryptography (methods and implementations) are acceptable for key management (i.e.; generation, access, distribution, destruction, handling, and storage of keys) and cryptographic services (i.e.; encryption, decryption, signature, hashing, key distribution, and random number generation services). | O.CRYPTOGRAPHY_VALIDATED<br>The TOE shall use NIST FIPS 140-2 validated cryptomodules for cryptographic services implementing FIPS-approved security functions and random number generation services used by cryptographic functions. | O.CRYPTOGRAPHY_VALIDATED (FCS_BCM_(EXP).1, FCS_CKM.1(1), FCS_CKM.1(2)) satisfies this policy by requiring the TOE to implement NIST FIPS validated cryptographic services. These services will provide confidentiality and integrity protection of TSF data while in transit to remote parts of the TOE. |
| | O.RESIDUAL_INFORMATION<br>The TOE will ensure that any information contained in a protected resource is not released when the resource is reallocated or upon completion of a function that residual biometric data could not be reused. | O.RESIDUAL_INFORMATION (FDP_RIP.2, FCS_CKM.4) satisfies this policy by ensuring that cryptographic data are cleared from resources that are shared between users. Keys must be zeroized according to FIPS 140-2. |

| | **Objectives Addressing the Threat/Policy** | |
|---|---|---|
| P.VULNERABILITY_ANALYSIS_TEST<br>The TOE must undergo appropriate independent vulnerability analysis and penetration testing to demonstrate that the TOE is resistant to an attacker possessing a medium attack potential. | O.VULNERABILITY_ANALYSIS_ TEST<br>The TOE will undergo appropriate independent vulnerability analysis and penetration testing to demonstrate the design and implementation of the TOE does not allow attackers with medium attack potential to violate the TOE's security policies. | O.VULNERABILITY_ANALYSIS_TEST (AVA_VLA.3) satisfies this policy by ensuring that an independent analysis is performed on the TOE and penetration testing based on that analysis is performed. Having an independent party perform the analysis helps ensure objectivity and eliminates preconceived notions of the TOE's design and implementation that may otherwise affect the thoroughness of the analysis. The level of analysis and testing requires that an attacker with a moderate attack potential cannot compromise the TOE's ability to enforce its security policies. |

## 6.2 Rationale for the Security Objectives and Security Functional Requirements for the Environment

190 This PP does not require any security requirements on the IT environment; therefore no additional rationale is needed here.

## 6.3 Rationale for TOE Security Requirements

**Table 10 Rationale for TOE Security Requirements**

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| O.ADMIN_ROLE<br>The TOE will provide administrator roles to isolate administrative actions, and to make the administrative functions available locally and remotely. | FMT_SMR.2 | FMT_SMR.2 requires that three roles exist for administrative actions: the Security Administrator, who is responsible for configuring most security-relevant parameters on the TOE; the Cryptographic Administrator, who is responsible for managing the security data that is critical to the cryptographic operations; and the Audit Administrator, who is responsible for reading and deleting the audit trail. The TSF is able to associate a human user with one or more roles and these roles isolate administrative functions in that the functions of these roles do not overlap. It is true that the design of some systems could enable a rogue security administrator to manipulate cryptographic data by, for instance, writing directly to kernel memory. While this scenario is a security concern, this objective does not counter that aspect of T.ADMIN_ROGUE. If a security administrator were to perform such an action, the auditing requirements (along with the audit trail protection requirements) afford some measure of detectability of the rogue administrator's actions. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| O.AUDIT_GENERATION<br>The TOE will provide the capability to detect and create records of security-relevant events associated with users. | FAU_GEN.1-NIAP-0410 | FAU_GEN.1-NIAP-0407 defines the set of events that the TOE must be capable of recording. This requirement ensures that an administrator has the ability to audit any security relevant event that takes place in the TOE. This requirement also defines the information that must be contained in the audit record for each auditable event. There is a minimum of information that must be present in every audit record and this requirement defines that, and the additional information that must be recorded for each auditable event. This requirement also places a requirement on the level of detail that is recorded on any additional security functional requirements an ST author adds to this PP. |
| | FAU_GEN.2-NIAP-0410 | FAU_GEN.2-NIAP-0410 ensures that the audit records associate a user identity with the auditable event. Although the FIA_ATD.1 requirements mandate that a "userid" be used to represent a user identity, the TOE developer is able to associate different types of user-ids with different users in order to meet this objective. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | FIA_USB.1 | FIA_USB.1 plays a role is satisfying this objective by requiring a binding of security attributes associated with users that are authenticated with the subjects that represent them in the TOE. This only applies to authenticated users, since the identity of unauthenticated users cannot be confirmed. Therefore, the audit trail may not always have the proper identity of the subject that causes an audit record to be generated. |
| | FAU_SEL.1 – NIAP- 0407 | FAU_SEL.1-NIAP-0407 allows the selected administrator(s) to configure which auditable events will be recorded in the audit trail. This provides the administrator with the flexibility in recording only those events that are deemed necessary by site policy, thus reducing the amount of resources consumed by the audit mechanism and providing the ability to focus on the actions of an individual user. In addition, the requirement has been refined to require that the audit event selection function is configurable during run-time to ensure the TOE is able to capture security-relevant events given changes in threat conditions. |
| O.AUDIT_PROTECTION<br>The TOE will provide the capability to protect audit information. | FMT_MOF.1(2)<br>FAU_SAR.2<br>FAU_STG.1-NIAP-0429<br>FAU_STG.3<br>FAU_STG.NIAP-0414-1-NIAP-0429<br>FMT_SMF.1 | FMT_MOF.1 restricts the ability to control the behavior of the audit and alarm mechanism to the Security Administrator. The Security Administrator is the only user that controls the behavior of the events that generate alarms and whether the alarm mechanism is enabled or |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | disabled.<br><br>FAU_SAR.2 restricts the ability to read the audit trail to the Auditor, thus preventing the disclosure of the audit data to any other user. However, the TOE is not expected to prevent the disclosure of audit data if it has been archived or saved in another form (e.g., moved or copied to an ordinary file).<br><br>The FAU_STG family dictates how the audit trail is protected. FAU_STG.1-NIAP-0429 restricts the ability to delete audit records to the Audit Administrator; or if the option of overwriting old audit records is chosen by the Administrator in FAU_STG.NIAP-0414-1-NIAP-0429, the audit data may be deleted/overwritten. Since the Audit Administrator is trusted to review the audit data, the threat being countered is that the administrator does something malicious and then attempts to conceal it by configuring the audit log to overwrite old records. Presumably the administrator would then attempt to fill up the audit log in order to overwrite the thing they just did, and the fact that they reconfigured the audit log overwrite action. The Audit Administrator would hopefully notice this activity and detect the fact that the administrator was performing illicit activities. The fact that the administrator does not directly have the ability to delete the audit records helps ensure that audit records are |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | kept until the Audit Administrator deems they are no longer necessary. FAU_STG.1-NIAP-0429 also ensures that no one has the ability to modify audit records (e.g., edit any of the information contained in an audit record). This ensures the integrity of the audit trail is maintained.<br><br>FAU_STG.3 requires that the administrators be alerted when the audit trail exceeds a capacity threshold established by the Security Administrator. In addition, an audit record is cut which will trigger the analysis performed in FAU_SAA, resulting in an FAU_ARP alarm being issued. This ensures that an administrator has the opportunity to manage the audit trail before it becomes full and the avoiding the possible loss of audit data.<br><br>FAU_STG.NIAP-0414-1-NIAP-0429 allows the Security Administrator to configure the TOE so that if the audit trail does become full, either the TOE will prevent any events from occurring (other than actions taken by the administrator) that would generate an audit record or the audit mechanism will overwrite the oldest audit records with new records.<br><br>FMT_SMF.1 requires the TOE to provide an administrator with a facility to backup, recover and archive audit data ensuring the ability to recover corrupted audit records, and access to a complete |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | history of audit information. |
| O.AUDIT_REVIEW<br>The TOE will provide the capability to selectively view audit information, and alert the administrator of identified potential security violations. | FAU_ARP.1<br>FAU_ARP_ACK_(EXP).1<br>FAU_SAA.1-NIAP-0407<br>FAU_SAR.1<br>FAU_SAR.3 | FAU_SAA.1-NIAP-0407 defines the events (or rules) that indicate a potential security violation and will generate an alarm. The triggers for these events are largely configurable by the Security Administrator. Some rules are not configurable, or configurable by the cryptographic administrator.<br>FAU_ARP.1 requires that the alarm be displayed at the local administrative console and at the remote administrative console(s) when auditor and security administrative session(s) exists. For alarms at remote consoles, the alarm is sent either during an established session or upon session establishment (as long as the alarm has not been acknowledged). This is required to increase the likelihood that the alarm will be received as soon as possible. This requirement also dictates the information that must be displayed with the alarm. The potential security violation is identified in the alarm, as are the contents of the audit records of the events that accumulated and triggered the alarm. The information in the audit records is necessary; it allows the administrators to react to the potential security violation without having to search through the audit trail looking for the related events. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | FAU_ARP_ACK_(EXP).1 requires that an alarm generated by the mechanism that implements the FAU_ARP requirement be maintained until an administrator acknowledges it. This ensures that the alarm message will not be obstructed and the administrators will be alerted of a potential security violation. Additionally, this requires that the acknowledgement be transmitted to users that received the alarm, thus ensuring that that set of administrators knows that the user specified in the acknowledgement message has addressed the alarm.<br><br>FAU_SAR.1 (both iterations) is used to provide both the auditor and an external audit analysis function the capability to read the entire audit data contained in the audit trail. This requirement also mandates the audit information be presented in a manner that is suitable for the end user (auditor or external system) to interpret the audit trail. It is expected that the audit information be presented in such a way that the end user can examine an audit record and have the appropriate information (that required by FAU_GEN.2-NIAP-410) presented together to facilitate the analysis of the audit review. Ensuring the audit data are presented in an interpretable format will enhance the ability of the |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | entity performing the analysis to identify potential security violations.<br><br>FAU_SAR.3 complements FAU_SAR.1 by providing the administrators the flexibility to specify criteria that can be used to search or sort the audit records residing in the audit trail. FAU_SAR.3 requires the administrators be able to establish the audit review criteria based on a userid and role so that the actions of a user can be readily identified and analyzed. Allowing the administrators to perform searches or sort the audit records based on dates and times provides the capability to facilitate the administrator's review of incidents that may have taken place at a certain time. It is important to note that the intent of sorting in this requirement is to allow the administrators the capability to organize or group the records associated with a given criteria. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| O.CHANGE_MANAGEMENT<br>The configuration of, and all changes to, the TOE and its development evidence will be analyzed, tracked, and controlled throughout the TOE's development. | ACM_CAP.4<br>ACM_SCP.2<br>ALC_DVS.1<br>ALC_FLR.2<br>ALC_LCD.1<br>ACM_AUT.1 | ACM_CAP.4 contributes to this objective by requiring the developer have a configuration management plan that describes how changes to the TOE and its evaluation deliverables are managed. The developer is also required to employ a configuration management system that operates in accordance with the CM plan and provides the capability to control who on the development staff can make changes to the TOE and its developed evidence. This requirement also ensures that authorized changes to the TOE have been analyzed and the developer's acceptance plan describes how this analysis is performed and how decisions to incorporate the changes to the TOE are made<br>ACM_SCP.2 is necessary to define what items must be under the control of the CM system. This requirement ensures that the TOE implementation representation, design documentation, test documentation (including the executable test suite), user and administrator guidance, CM documentation and security flaws are tracked by the CM system. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | ALC_DVS.1 requires the developer describe the security measures they employ to ensure the integrity and confidentiality of the TOE is maintained. The physical, procedural, and personnel security measures the developer uses provides an added level of control over who and how changes are made to the TOE and its associated evidence.<br><br>ALC_FLR.2 plays a role in satisfying the "analyzed" portion of this objective by requiring the developer to have procedures that address flaws that have been discovered in the product, either through developer actions (e.g., developer testing) or those discovered by others. The flaw remediation process used by the developer corrects any discovered flaws and performs an analysis to ensure new flaws are not created while fixing the discovered flaws.<br><br>ALC_LCD.1 requires the developer to document the life-cycle model used in the development and maintenance of the TOE. This life-cycle model describes the procedural aspects regarding the development of the TOE, such as design methods, code or documentation reviews, how changes to the TOE are reviewed and accepted or rejected. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | ACM_AUT.1 complements ACM_CAP.4, by requiring that the CM system use an automated means to control changes made to the TOE. If automated tools are used by the developer to analyze, or track changes made to the TOE, those automated tools must be described. This aids in understanding how the CM system enforces the control over changes made to the TOE. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| O.CORRECT_ TSF_OPERATION<br>The TOE will provide the capability to test the TSF to ensure the correct operation of the TSF in its operational environment. | FPT_TST_(EXP).4, FPT_TST_(EXP).5 | O_CORRECT_TSF_OPERATION requires two security functional requirements in the FPT class, FPT_TST. These functional requirements provide the end user with the capability to ensure the TOE's security mechanisms continue to operate correctly in the field. FPT_TST_(EXP).4 has been created to ensure end user tests exist to demonstrate the correct operation of the security mechanisms required by the TOE that are provided by the hardware and that the TOE's software and TSF data has not been corrupted. Hardware failures could render a TOE's software ineffective in enforcing its security policies and this requirement provides the end user the ability to discover any failures in the hardware security mechanisms. FPT_TST_(EXP).4 is necessary to ensure the correctness of the TSF software and TSF data. If TSF software is corrupted it is possible that the TSF would no longer be able to enforce the security policies. This also holds true for TSF data, if TSF data is corrupt the TOE may not correctly enforce its security policies. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| O.CRYPTOGRAPHY_VALIDATED<br>The TOE shall use NIST FIPS 140-2 validated cryptomodules for cryptographic services implementing FIPS-approved security functions and random number generation services used by cryptographic functions. | FCS_BCM_(EXP).1<br>FCS_CKM.1(1)<br>FCS_CKM.1(2) | This objective deals with the issue of using FIPS 140-2-approved cryptomodules in the TOE. A cryptomodule, as used in the components, is a module that is FIPS 140-2 validated (in accordance with FCS_BCM_(EXP).1); the cryptographic functionality implemented in that module are FIPS-approved security functions that have been validated; and the cryptographic functionality is available in a FIPS-approved mode of the cryptomodule. This objective is distinguished from O.CRYPTOGRAPHIC_FUNCTIONS in that this deals only with a requirement to use FIPS 140-2-validated cryptomodules where the TOE requires such functionality; it does not dictate the specific functionality that is to be used.<br>FCS_BCM_(EXP).1 is an explicit requirement that specifies not only that cryptographic functions that are FIPS-approved and must be validated by FIPS, but also what NIST FIPS rating level the cryptographic module must satisfy. The level specifies the degree of testing of the module. The higher the level, the more extensive the module is tested. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | FCS_CKM.1(1) and FCS_CKM.1(2) mandate that the cryptomodule must generate symmetric and asymmetric keys, and that this key generation must be by one of the specified methods. |
| O.CRYPTOGRAPHIC_FUNCTIONS<br>The TOE shall provide cryptographic functions (i.e., encryption/decryption and digital signature operations) to maintain the confidentiality and allow for detection of modification of TSF data that is transmitted between physically separated portions of the TOE, or stored outside the TOE. | FCS_CKM.1(1)<br>FCS_CKM.1(2)<br>FCS_CKM.2<br>FCS_CKM.4<br>FCS_CKM_(EXP).1<br>FCS_CKM_(EXP).2<br>FCS_COA_(EXP).1<br>FCS_COP.1(1)<br>FCS_COP.1(2)<br>FCS_COP.1(3)<br>FCS_COP.1(4)<br>FCS_COP.1(5) | The FCS requirements used in this PP satisfy this objective by levying requirements that ensure the cryptographic standards include the NIST FIPS publications (where possible) and NIST approved ANSI standards. The intent is to have the satisfaction of the cryptographic standards be validated through a NIST FIPS 140 validation.<br>In contrast to O.CRYPTOGRAPHY_VALIDATED, this objective is to provide cryptographic functionality that is used by the TOE. The core functionality to be supported is encryption/decryption using a symmetric algorithm, and digital signature generation and verification using asymmetric algorithms. Since these operations involve cryptographic keys, how the keys are generated and/or otherwise obtained have to also be specified.<br>FCS_CKM.1(1) is a requirement that a cryptomodule generate symmetric |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | keys. Such keys are used by the TDEA or AES encryption/decryption functionality specified in FCS_COP.1(1).<br><br>FCS_CKM.1(2) is a requirement that a cryptomodule generate asymmetric keys. Such keys are used for cryptographic signatures as specified in FCS_COP.1(2).<br><br>FCS_CKM_(EXP).1 requires that the TSF validate all keys generated to assure that it meet relevant standards.<br><br>FCS_CKM_(EXP).2 requires that keys are handled appropriately and associated with the correct entities, and that transfer of keys is done with error detection. Storage of persistent secret and private keys must be done in a secure fashion.<br><br>FCS_COA_(EXP).1 requires the TSF to provide encryption, decryption, digital signature, key agreement and secure hashing services.<br><br>FCS_COP.1(3) requires that the TSF provide hashing services using a NIST-approved implementation of the Secure Hash Algorithm and FCS_COP.1(4) requires the TSF's message authentication services be compliant with either of the NIST-approved approaches, HMAC or CCM..<br><br>Another way of obtaining key material for symmetric algorithms is through cryptographic key establishment, as specified in |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | FCS_COP.1(5). Key establishment has two aspects: key agreement and key distribution. Key agreement occurs when two entities exchange public data yet arrive at a mutually shared key without ever passing that key between the two entities (for example, the Diffie-Hellman algorithm).<br><br>Key distribution (FCS_CKM.2) occurs when the key is transmitted from one entity to the TOE. If the entity is electronic and a protocol is used to distribute the key, it is referred to in this PP as "Key Transport". If the key is loaded into the TOE it can be loaded electronically (e.g., from a floppy drive, smart card, or electronic keyfill device) or manually (e.g., typed in). One or more of these methods must be selected.<br><br>FCS_CKM.4 provides the functionality for ensuring key and key material is zeroized. This applies not only to key that resides in the TOE, but also to intermediate areas (physical memory, page files, memory dumps, etc.) where key may appear.<br><br>FCS_COP.1(1) specifies that TDEA or AES be used to perform encryption and decryption operations. FCS_COP.1(2) gives three options for providing the digital signature capability; these requirements reference the appropriate standards for each digital signature option.. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| O.DISPLAY_BANNER<br>The TOE will display an advisory warning regarding use of the TOE. | FTA_TAB.1 | FTA_TAB.1 meets this objective by requiring the TOE display a Security Administrator defined banner before a user can establish an authenticated session. This banner is under complete control of the Security Administrator in which they specify any warnings regarding unauthorized use of the TOE and remove any product or version information if they desire. |
| O.DOCUMENT_KEY_ LEAKAGE<br>The bandwidth of channels that can be used to compromise key material shall be documented. | AVA_CCA_(EXP). 2 | AVA_CCA_(EXP).2 requires that a covert channel analysis be performed on the entire TOE to determine the bandwidth of possible cryptographic key leakage. While there are no requirements to limit the bandwidth, the results of this analysis will provide useful guidance on what the specified lifetime of the cryptographic keys should be in order to reduce the damage due to a key compromise. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| O.MAINT_MODE<br>The TOE shall provide a mode from which recovery or initial startup procedures can be performed. | FPT_RCV.2 | This objective is met by using the FPT_RCV.2 requirement, which ensures that the TOE does not continue to operate in an insecure state when a hardware or software failure occurs. Upon the failure of the TSF self-tests the TOE will no longer be assured of enforcing its security policies. Therefore, the TOE enters a state that operations cease and requires an administrator to follow documented procedures that instruct them on to return the TOE to a secure state. These procedures may include running diagnostics of the hardware, or utilities that may correct any integrity problems found with the TSF data or code. Solely specifying that the administrator reload and install the TOE software from scratch, while may be required in some cases, does not meet the intent of this requirement. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| O.MANAGE<br>The TOE will provide all the functions and facilities necessary to support the administrators in their management of the security of the TOE, and restrict these functions and facilities from unauthorized use. | FMT_MSA.1(1)<br>FMT_MSA.1(2)<br>FMT_MSA.3(1)<br>FMT_MSA.3(2)<br>FMT_MOF.1(1)<br>FMT_MOF.1(2)<br>FMT_MOF.1(3)<br>FMT_MOF.1(4)<br>FMT_MOF.1(5)<br>FMT_MOF.1(6)<br>FMT_MTD.1(1)<br>FMT_MTD.1(2)<br>FMT_MTD.1(3)<br>FMT_MTD.1(4)<br>FMT_SMF.1 | The FMT requirements are used to satisfy this management objective, and other objectives that specify the control of functionality. The requirement's rationale for this objective focuses on the administrator's capability to perform management functions in order to control the behavior of security functions.<br>FMT_MSA.1(1) and FMT_MSA.1(2) both provide the Security Administrator the capability to manipulate the security attributes of the objects in their scope of control that determine the access policy.<br>FMT_MSA.3(1) requires that by default, the TOE does not allow an information flow, rather than allowing information flows until a rule in the ruleset disallows it.<br>FMT_MOF.1(2) and FMT_MSA.3(2) are related to the services provided by FAU_UAU.1(1) and provide the Security Administrator control as to the availability of these services. FMT_MOF.1(2) provides the ability to enable or disable the TOE services to the Security Administrator.<br>FMT_MSA.3(2) requires that these services by default are disabled. Since the Security Administrator must explicitly enable these services it |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | ensures the Security Administrator is aware that they are running. This requirement does afford the Security Administrator the capability to override this restrictive default and allow the services to be started whenever the TOE reboots or is restarted.<br><br>FMT_MOF.1(1) is used to ensure the administrators have the ability to invoke the TOE self-tests at any time. The ability to invoke the self-tests is provided to all administrators. The Security Administrator is able to modify the behavior of the tests (e.g., select when they run, select a subset of the tests).<br><br>FMT_MOF.1(3) specifies the ability of the administrators to control the security functions associated with audit and alarm generation. The ability to control these functions has been assigned to the appropriate administrative roles.<br><br>FMT_MOF.1(6) This requirement limits the ability to manipulate the values that are used in the FRU_RSA.1(2) requirements to the Security Administrator. The Security Administrator is provided the capability to assign the network identifier(s) they wish to place resource restrictions on and allows them to also specify over what period |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | of time those quota limitations are in place.<br><br>FMT_MOF.1(4) provides the administrators "read only" access to the audit records and prohibits access to all other users. Additionally, the administrators are provided the capability to "search and sort" audit on defined criteria. This capability expedites problem resolution analysis.<br><br>FMT_MOF.1(5) ensures that only an administrators can "enable or disable" the security alarms. This requirement works with FMT_MOF.1(5) to provide detailed granularity to the administrator when determining which actions constitute a security violation.<br><br>FMT_MOF.1(6) provides the Security Administration configuration control of the allocation of connection-oriented TOE resources. This requirement provides the Security Administrator with a capability to thwart possible external "resource allocation" attacks on the TOE.<br><br>The requirement FMT_MTD.1(1) is intended to be used by the ST author, with possible iterations, to address TSF data that has not already been specified by other FMT requirements. This is necessary because the ST author may add TSF data in assignments that cannot be addressed ahead of time by the PP authors. This |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | requirement specifies that the manipulation of these data be restricted to the security administrator.<br><br>FMT_MTD.1(2) provides the Cryptographic Administrator, and only the Cryptographic Administrator, the ability to modify the cryptographic security data. This allows the Cryptographic Administrator to change the critical data that affects the TOE's ability to perform its cryptographic functions properly.<br><br>FMT_MTD.1(3) provides the capability of setting the date and time that is used to generate time stamps to the Security Administrator or a trusted IT entity (authorized data manager). It is important to allow this functionality, due to clock drift and other circumstances, but the capability must be restricted. A trusted IT entity is allowed in the selection made by the ST author to take in account the use of an NTP server or some other service that provides time information without human intervention.<br><br>FMT_MTD.1(4) addresses the capabilities of data managers, who have responsibilities for security data management for sub-portions of the set of TSF data (for example, the platform clock time, sub-hierarchies |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | of the directory). The scope of a data manager's responsibility is set by a security administrator, but they are expected to manage the entities in their scope of control without relianceon the security administrator.<br><br>FMT_MTD.2(1), FMT_MTD.2(2) restrict the setting of limits on the processor time and network connection resources, respectively, to an administrator. This capability allows an administrator to control the resources consumed by, to provide a flexible policy with respect to denial of service attacks.<br><br>The requirement FMT_SMF.1 was introduced as an international interpretation. This requirement specifies functionality that must be provided to administrators of the TOE. If the PP author includes this requirement, care must be taken to use the other FMT requirements to specify how the functionality is restricted and to which role the functionality is provided. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| O.MEDIATE_INFORMATION_FLOW<br>The TOE must mediate the flow of information between sets of TOE network interfaces or between a network interface and the TOE itself in accordance with its security policy. | FDP_IFC.1(1)<br>FDP_IFC.1(2)<br>FDP_IFF.1(1)<br>FDP_IFF.1(2)<br>FMT_REV.1<br>FPT_RVM.1 | The FDP_IFF and FDP_IFC requirements were chosen to define the policies, the subjects, objects, and operations for how and when mediation takes place.<br>FDP_IFC.1(1), and FDP_IFC.1(2) define the subjects, information (e.g., objects) and the operations that are performed with respect to the two information flow policies.<br>FDP_IFC.1(1) defines subjects for the unauthenticated access to any services the TOE provides. This is different from the other policies in that the TOE mediates access to itself, rather than determining if information should be allowed to flow through the TOE. The destination subject is defined to be the TOE, and the source subject is the TOE interface on which a network packet is received. The information remains the same, a network packet, and the operations are limited to accept or reject the packet.<br>FDP_IFF.1(1) provides the rules that apply to the unauthenticated use of any services provided by the TOE. ICMP is the only service that is required to be provided by the TOE, and the security attributes associated with this protocol allow the Security Administrator to specify what degree the ICMP traffic is mediated (i.e., the ICMP message type and code). |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | The ST author could specify other services they wish their TOE implementation to provide, and if they do so, they should also specify the security attributes associated with the additional services. FMT_REV.1 is a management requirement that affords the Security Administrator the ability to immediately revoke user's ability to send network traffic to or through the TOE.<br><br>FDP_IFC.1(2), the subjects are the TOE's network interfaces. The objects are defined as the network IP packets on which the TOE performs routing operations. As packets enter the TOE, the network interface where they are received is the source subject. As packets are sent out of the TOE the network interface that they are sent out of is the destination subject. Subjects must be defined as entities that the TOE has control over. The TOE has control over its own network interfaces such that it can make information flow decisions to allow/disallow network packets to flow from in incoming interface to an outgoing interface, and can apply routing operations to packets that are allowed to flow. To define subjects as the senders and receivers of network packets would not allow specification of an information flow policy that the TOE could enforce, since the sender |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | and receiver of network packets are not under the contol of the TOE. The operations defined are those of the routing policy. The routing policy either passes information along or denies the information flow.<br><br>FDP_IFF.1(2) specifies the attributes on which authenticated information flow decisions are made. Each TOE interface has a set of source subject identifiers that is the list of senders of information packets that are allowed to send packets to this TOE interface. Each TOE interface also has a list of destination subject identifiers that specifies the receivers that network packets can be sent to on that TOE interface. As packets are received on a particular network interface, the TOE determines if they are allowed to enter on that interface. Then based on rules defined by the Security Administrator, the TOE applies authenticated routing operations to the packet. Before the packet is sent out of a particular network interface, the TOE determines if the destination (i.e., receiver) of the packet is in the list of destinations that may be reached over that interface.<br><br>FMT_REV.1 is a management requirement that affords the Security Administrator the ability to immediately revoke user's ability to send network traffic to or through |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | the TOE.<br>If the Security Administrator revokes a user's access (e.g., via a rule in the ruleset, revoking an administrative role from a user) the TOE will immediately enforce the new Security Administrator defined "policy". FPT_RVM.1 ensures that packets that flow through the TOE, or those that are destined for the TOE are mediated with respect to the identified policies. Each TSF interface that operates on subjects or objects that are identified in the explicit policies, or operates on TSF data or security attributes, must ensure that the operation is checked against the explicit and implicit security policies defined in this PP. If any TSF interface allows unchecked access to any of these resources, then the TOE cannot be relied upon to enforce the security policies. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| O.PEER_AUTHENTICATION | FCS_IKE_(EXP).1 | The O.PEER_AUTHENTICATION objective is satisfied by the requirement FCS_IKE_(EXP).1, which specifies that the TOE must implement the Internet Key Exchange protocol defined in RFC 2409. By implementing this protocol, the TOE will establish a secure, authenticated channel with each peer TOE for purposes of establishing a security association, which includes the establishment of a cryptographic key, algorithm and mode to be used for all communication. It is possible to establish multiple security associations between two peer TOEs, each with its own cryptographic key. Authentication may be via a digital signature or pre-shared key. |
| O.PROTECT_IN_TRANSIT<br>The TSF shall protect TSF data when it is in transit between the TSF and another trusted IT entity. | FPT_ITA.1<br>FPT_ITC.1<br>FPT_ITI.1<br>FTP_ITC.1(1)<br>FTP_ITC.1(2)<br>FTP_TRP.1(1)<br>FTP_TRP.1(2) | FPT_ITA.1, FPT_ITC.1 and FPT_ITI.1 are concerned with the availability, confidentiality and integrity of the TSF data while being transmitted.<br>FTP_ITC.1(1) and FTP_ITC.1(2) ensures that all TSF data will be protected from disclosure while in transit from the TOE to another trusted IT entity.<br>FTP_TRP.1(1) and FTP_TRP.1(2) will use cryptographic means to provide prevention of disclosure and detection of modification of TSF data. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| O.PROTOCOLS<br>The TOE will ensure that standardized protocols are implemented in the TOE to RFC and/or Industry specifications to ensure interoperability. | FPT_FLS.1<br>FPT_PRO_(EXP).1 | The O.PROTOCOLS objective is satisfied by FPT_PRO_(EXP).1, which requires that the TOE be implemented with standardized protocols to ensure interoperability among peer TOEs. Implementing the standardized protocols will ensure that a secure state (FPT_FLS.1) of the TOE is maintained. |
| O.REPLAY_DETECTION<br>The TOE will provide a means to detect and reject the replay of TSF data and security attributes. | FPT_RPL.1 | The O.REPLAY_DETECTION objective is satisfied by FPT_RPL.1, which requires the TOE to detect and reject the attempted replay of authentication data from a remote user. This is sufficient to meet the objective because no untrusted users have local access to the TOE, thus there is no way to capture neither replay authentication data for a local session. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| O.RESIDUAL_INFORMATION<br>The TOE will ensure that any information contained in a protected resource is not released when the resource is reallocated. | FDP_RIP.2<br>FCS_CKM.4<br>FCS_CKM_(EXP).2 | FDP_RIP.2 is used to ensure the contents of resources are not available to subjects other than those explicitly granted access to the data. For this TOE it is critical that the memory used to build network packets is either cleared or that some buffer management scheme be employed to prevent the contents of a packet being disclosed in a subsequent packet (e.g., if padding is used in the construction of a packet, it must not contain another user's data or TSF data).<br>FCS_CKM.4 applies to the destruction of cryptographic keys used by the TSF. This requirement specifies how and when cryptographic keys must be destroyed. The proper destruction of these keys is critical in ensuring the content of these keys cannot possibly be disclosed when a resource is reallocated to a user. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| O.RESOURCE_SHARING<br>The TOE shall provide mechanisms that mitigate attempts to exhaust connection-oriented resources provided by the TOE (e.g., entries in a connection state table; Transmission Control Protocol (TCP) connections to the TOE). | FRU_RSA.1(1)<br>FRU_RSA.1(2)<br>FMT_MTD.2(1)<br>FMT_MTD.2(2)<br>FMT_MOF.1 (6) | While an availability security policy does not explicitly exist, FRU_RSA.1 was used to mitigate potential resource exhaustion attempts. FRU_RSA.1(1) was used to reduce the impact of an attempt being made to exhaust the transport-layer representation (e.g., attempt to make the TSF unable to respond to connection-oriented requests, such as SYN attacks). This requirement allows the administrator to specify the time period in which when maximum quota (which is defined by the ST) is met or surpassed, an ST defined action is to take place, which is specified in FMT_MTD.2(1). These two requirements together help limit the resources that can be utilized by the general population of users as a whole. An issue with treating all the users the same is that legitimate users may not be able to establish connections due to the connection table entries being exhausted. Therefore FRU_RSA.1(2) is also included.<br>FRU_RSA.1(2) is more specific in that attempts to exhaust the connection-oriented resources by a single network address, or a set of network addresses can be controlled. This affords the administrator a finer granularity of control than FRU_RSA.1(1). |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | FRU_RSA.1(2) has the advantage of providing the Security Administrator with the ability to define the maximum number of resources a particular address or set of addresses can use over a specified time period. This requirement works in conjunction with FMT_MTD.2(2) which restricts the ability to set the quotas to the security administrator and allows for the ST author to assign what actions will take place once the quotas are met or surpassed. This iteration of FPT_RSA.1 makes it less likely that a legitimate user of the TOE will be denied access due to resource exhaustion attempts.<br><br>FMT_MOF.1(6) restricts the ability to assign the single network address or set of network addresses used in FRU_RSA.1(2) to the Security Administrator. This is in keeping with the TOE's notion of the Security Administrator is responsible for configuring the TOE's policy enforcement mechanisms. |
| O.ROBUST_ADMIN_GUIDANCE<br><br>The TOE will provide administrators with the necessary information for secure delivery and management. | ADO_DEL.2<br><br>ADO_IGS.1<br><br>AGD_ADM.1<br><br>AGD_USR.1<br><br>AVA_MSU.1 | ADO_DEL.2 ensures that the administrator is provided documentation that instructs them how to ensure the delivery of the TOE, in whole or in parts, has not been tampered with or corrupted during delivery. This requirement ensures the administrator has the ability to begin their TOE installation with a clean (e.g., malicious code has not been inserted once it has left the developer's control) version of the TOE, which is |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | necessary for secure management of the TOE.<br><br>The ADO_IGS.1 requirement ensures the administrator has the information necessary to install the TOE in the evaluated configuration. Often times a vendor's product contains software that is not part of the TOE and has not been evaluated. The Installation, Generation and Startup (IGS) documentation ensures that once the administrator has followed the installation and configuration guidance the result is a TOE in a secure configuration.<br><br>The AGD_ADM.1 requirement mandates the developer provide the administrator with guidance on how to operate the TOE in a secure manner. This includes describing the interfaces the administrator uses in managing the TOE, security parameters that are configurable by the administrator, how to configure the TOE's ruleset and the implications of any dependencies of individual rules. The documentation also provides a description of how to setup and review the auditing features of the TOE.<br><br>The AGD_USR.1 is intended for non-administrative users, but could be used to provide guidance on security that is common to both administrators and non-administrators (e.g., password management guidelines).<br><br>AVA_MSU.2 ensures that the guidance documentation is complete |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | and can be followed unambiguously to ensure the TOE is not mis-configured in an unsecure state due to confusing guidance. |
| O.ROBUST_TOE_ACCESS<br>The TOE will provide mechanisms that control a user's logical access to the TOE and to explicitly deny access to specific users when appropriate | FTA_TSE.1<br>FIA_UID.2<br>FTA_SSL.3<br>AVA_SOF.1<br>FIA_AFL.1<br>FIA_ATD.1(1)<br>FIA_ATD.1(2)<br>FIA_UAU.2<br>FIA_UAU_(EXP).5 | FIA_UID.2 plays a small role in satisfying this objective by ensuring that every user is identified before the TOE performs any mediated functions. FIA_ATD.1(1) defines the attributes of users, including a userid that is used to by the TOE to determine a user's identity and enforce what type of access the user has to the TOE (e.g., the TOE associates a userid with any role(s) they may assume). This requirement allows a human user to have more than one user identity assigned, so that a single human user could assume all the roles necessary to manage the TOE. In order to ensure a separation of roles, this PP requires a single role to be associated with a user id. This is inconvenient in that the administrator would be required to log in with a different user id each time they wish to assume a different role, but this helps mitigate the risk that could occur if an administrator were to execute malicious code.<br>FIA_ATD.1(2) defines the attributes of IT entities, including a subject ID that is used to by the TOE to determine an entity's identity and enforce what type of access the entity has to the TOE. This requirement allows an IT entity to have more than one subject identity assigned, so that a single entity (e.g., |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | another router) could assume the necessary role required to manage the TOE (e.g updating the routing tables).<br><br>FIA_UAU.2 requires that administrators and authorized IT entities authenticate themselves to the TOE before performing any TSF-mediated actions. In order to control logical access to the TOE an authentication mechanism is required. The explicit requirement FIA_UAU_(EXP).5 mandates that the TOE provide a local authentication mechanism. This requirement also affords the ST author the opportunity to add additional authentication mechanisms (e.g., single-use, certificates) if they desire.<br><br>Local authentication is required to ensure someone that has physical access to the TOE and has not been granted logical access (e.g., a janitor) cannot gain unauthorized logical access to the TOE.<br><br>The AVA_SOF.1 requirement is applied to the local authentication mechanism. For this TOE, the strength of function specified is medium. This requirement ensures the developer has performed an analysis of the authentication mechanism to ensure the probability of guessing a user's authentication data would require a high-attack potential, as defined in Annex B of the CEM.<br><br>FTA_TSE.1.1 contributes to this |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | objective by limiting a user's ability to logically access the TOE. This requirement provides the Security Administrator the ability to control when (e.g., time and day(s) of the week) and where (e.g., from a specific network address) remote administrators, as and authorized IT entities can access the TOE.<br><br>FIA_AFL.1 provides a detection mechanism for unsuccessful authentication attempts by remote administrators, and authorized IT entities. The requirement enables a Security Administrator settable threshold that prevents unauthorized users from gaining access to authorized user's account by guessing authentication data by locking the targeted account until the Security Administrator takes some action (e.g., re-enables the account) or for some Security Administrator defined time period. Thus, limiting an unauthorized user's ability to gain unauthorized access to the TOE.<br><br>The FTA_SSL family partially satisfies the O.ROBUST_TOE_ACCESS objective by ensuring that user's sessions are afforded some level of protection. FTA_SSL.3 takes into account remote sessions. After a Security Administrator defined time interval of inactivity remote sessions will be terminated. This includes user remote administrative sessions. This component is especially necessary; since remote sessions are not typically afforded the same |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | physical protections those local sessions are provided. |
| O.SELF_PROTECTION<br>The TSF will maintain a domain for its own execution that protects itself and its resources from external interference, tampering, or unauthorized disclosure. | FPT_SEP.2<br>FPT_RVM.1 | FPT_SEP.2 was chosen to ensure the TSF provides a domain that protects itself from untrusted users. If the TSF cannot protect itself it cannot be relied upon to enforce its security policies. FPT_SEP.1 could have been used to address the previous notion, however, FPT_SEP.2 was used to require that the cryptographic module be provided its own address space. This is necessary to reduce the impact of programming errors in the remaining portions of the TSF on the cryptographic module.<br>The inclusion of FPT_RVM.1 ensures that the TSF makes policy decisions on all interfaces that perform operations on subjects and objects that are scoped by the policies. Without this non-bypassability requirement, the TSF could not be relied upon to completely enforce the security policies, since an interface(s) may otherwise exist that would provide a user with access to TOE resources (including TSF data and executable code) regardless of the defined policies. This includes controlling the accessibility to interfaces, and what access control is provided within the interfaces. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| O.SOUND_DESIGN<br>The design of the TOE will be the result of sound design principles and techniques; the design of the TOE, and the design principles and techniques, are adequately and accurately documented. | ADV_ARC_(EXP).1<br>ADV_FSP_(EXP).1<br>ADV_HLD_(EXP).1<br>ADV_INT_(EXP).1<br>ADV_LLD_(EXP).1<br>ADV_RCR.1<br>ADV_SPM.1 | There are two different perspectives for this objective. One is from the developer's point of view and the other is from the evaluator's. The ADV class of requirements is levied to aide in the understanding of the design for both parties, which ultimately helps to ensure the design is sound.<br>ADV_ARC_(EXP).1 addresses the non-bypassability (FPT_RVM) and domain separation (FPT_SEP) aspects of the TSF, since these need to be analyzed differently from other functional requirements. The low-level design, as required by ADV_LLD_(EXP).1, provides the reader with the details of the TOE's design and describes at a module level how the design of the TOE addresses the SFRs. This level of description provides the detail of how modules interact within the TOE and if a flaw exists in the TOE's design, it is more likely to be found here rather than the high-level design. This requirement also mandates that the interfaces presented by modules be specified. Having knowledge of the parameters a module accepts, the errors that can be returned and a description of how the module works to support the security policies allows the design to be understood at its lowest level. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | ADV_INT_(EXP).1 ensures that the design of the TOE has been performed using good software engineering design principles that require a modular design of the TSF. Modular code increases the developer's understanding of the interactions within the TSF, which in turn, potentially reduces the amount of errors in the design. Having a modular design is imperative for evaluator's to gain an appropriate level of understanding of the TOE's design in a relatively short amount of time. The appropriate level of understanding is dictated by other assurance requirements in this PP (e.g., ATE_DPT.2, AVA_CCA_(EXP).2, AVA_VLA.3).<br><br>ADV_SPM.1 requires the developer to provide an informal model of the security policies of the TOE. Modeling these policies helps understand and reduce the unintended side-effects that occur during the TOE's operation that might adversely affect the TOE's ability to enforce its security policies. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | ADV_FSP_(EXP).1 requires that the interfaces to the TSF be completely specified. In this TOE, a complete specification of the network interface (including the network interface card) is critical in understanding what functionality is presented to untrusted users and how that functionality fits into the enforcement of security policies. Some network protocols have inherent flaws and users have the ability to provide the TOE with network packets crafted to take advantage of these flaws. The routines/functions that process the fields in the network protocols allowed (e.g., TCP, UPD, ICMP, any application level) must fully specified: the acceptable parameters, the errors that can be generated, and what, if any, exceptions exist in the processing. The functional specification of the hardware interface (e.g., network interface card) is also extremely critical. Any processing that is externally visible performed by NIC must be specified in the functional specification. Having a complete understanding of what is available at the TSF interface allows one to analyze this functionality in the context of design flaws. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | ADV_HLD_(EXP).1 requires that a high-level design of the TOE be provided. This level of design describes the architecture of the TOE in terms of subsystems. It identifies which subsystems are responsible for making and enforcing security relevant (e.g., anything relating to an SFR) decisions and provides a description, at a high level, of how those decisions are made and enforced. Having this level of description helps provide a general understanding of how the TOE works, without getting buried in details, and may allow the reader to discover flaws in the design.<br><br>The low-level design, as required by ADV_LLD_(EXP).1, provides the reader with the details of the TOE's design and describes at a module level how the design of the TOE addresses the SFRs. This level of description provides the detail of how modules interact within the TOE and if a flaw exists in the TOE's design, it is more likely to be found here rather than the high-level design. This requirement also mandates that the interfaces presented by modules be specified. Having knowledge of the parameters a module accepts, the errors that can be returned and a description of how the module works to support the security policies allows the design to |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | be understood at its lowest level.<br>The ADV_RCR.1 is used to ensure that the levels of decomposition of the TOE's design are consistent with one another. This is important, since design decisions that are analyzed and made at one level (e.g., functional specification) that are not correctly designed at a lower level may lead to a design flaw. This requirement helps in the design analysis to ensure design decisions are realized at all levels of the design. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| O.SOUND_IMPLEMENTATION<br>The implementation of the TOE will be an accurate instantiation of its design, and is adequately and accurately documented. | ADV_IMP.2<br>ADV_ARC_(EXP).1<br>ADV_LLD_(EXP).1<br>ADV_RCR.1<br>ADV_INT_(EXP).1<br>ALC_TAT.1 | While ADV_LLD_(EXP).1 is used to aide in ensuring that the TOE's design is sound, it also contributes to ensuring the implementation is correctly realized from the design. It is expected that evaluators will use the low-level design as an aide in understanding the implementation representation. The low-level design requirements ensure the evaluators have enough information to intelligently analyze (e.g., the documented interface descriptions of the modules match the entry points in the module, error codes returned by the functions in the module are consistent with those identified in the documentation) the implementation and ensure it is consistent with the design.<br>While evaluators have the ability to "negotiate" the subset in ADV_IMP.1, ADV_IMP.2 was chosen to ensure evaluators have full access to the source code. If the evaluators are limited in their ability to analyze source code they may not be able to determine the accuracy of the implementation or the adequacy of the documentation. Often times it is difficult for an evaluator to identify the complete sample of code they wish to analyze. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | Often times looking at code in one subsystem may lead the evaluator to discover code they should look at in another subsystem. Rather than require the evaluator to "re-negotiate" another sample of code, the complete implementation representation is required.<br><br>When performing the activities associated with the ADV_INT_(EXP).1 requirement, the evaluators will ensure that the architecture of the implementation is modular and consistent with the architecture presented in the low-level design. Having a modular implementation provides the evaluators with the ability to more easily assess the accuracy of the implementation, with respect to the design. If the implementation is overly complex (e.g., circular dependencies, not well understood coupling, reliance on side-effects) the evaluator may not have the ability to assess the accuracy of the implementation. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | ALC_TAT.1 provides evaluators with information necessary to understand the implementation representation and what the resulting implementation will consist of. Critical areas (e.g., the use of libraries, what definitions are used, compiler options) are documented so the evaluator can determine how the implementation representation is to be analyzed. |
| | | ADV_RCR.1 is used here to provide the correspondence of the lowest level of decomposition (e.g., source code) to the adjoining level, low-level design. The correspondence analysis is used by the evaluator as a tool when determining if the low-level design is correctly reflected in the implementation representation. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| O.THOROUGH_FUNCTIONAL_TESTING<br>The TOE will undergo appropriate security functional testing that demonstrates the TSF satisfies the security functional requirements. | ATE_COV.2<br>ATE_FUN.1<br>ATE_DPT.2<br>ATE_IND.2 | In order to satisfy O.FUNCTIONAL_TESTING, the ATE class of requirements is necessary. The component ATE_FUN.1 requires the developer to provide the necessary test documentation to allow for an independent analysis of the developer's security functional test coverage. In addition, the developer must provide the test suite executables and source code, which are used for independently verifying the test suite results and in support of the test coverage analysis activities. ATE_COV.2 requires the developer to provide a test coverage analysis that demonstrates the TSFI are completely addressed by the developer's test suite. While exhaustive testing of the TSFI is not required, this component ensures that the security functionality of each TSFI is addressed. This component also requires an independent confirmation of the completeness of the test suite, which aids in ensuring that correct security relevant functionality of a TSFI is demonstrated through the testing effort. ATE_DPT.2 requires the developer to provide a test coverage analysis that demonstrates depth of coverage of the test suite. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | This component complements ATE_COV.2 by ensuring that the developer takes into account the high-level and low-level design when developing their test suite. Since exhaustive testing of the TSFI is not required, ATE_DPT.2 ensures that subtleties in TSF behavior that are not readily apparent in the functional specification are addressed in the test suite. ATE_IND.2 requires an independent confirmation of the developer's test results, by mandating a subset of the test suite be run by an independent party. This component also requires an independent party to attempt to craft functional tests that address functional behavior that is not demonstrated in the developer's test suite. Upon successful adherence to these requirements, the TOE's conformance to the specified security functional requirements will have been demonstrated. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| O.TIME_STAMPS<br>The TOE shall provide reliable time stamps and the capability for the administrator to set the time used for these time stamps. | FPT_STM.1<br>FMT_MTD.1(3) | FPT_STM.1 requires that the TOE be able to provide reliable time stamps for its own use and therefore, partially satisfies this objective. Time stamps include date and time and are reliable in that they are always available to the TOE, and the clock must be monotonically increasing.<br>FMT_MTD.1(3) satisfies the rest of this objective by providing the capability to set the time used for generating time stamps to either the Security Administrator, authorized IT entity, or both, depending on the selection made by the ST author. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| O.TRUSTED_PATH<br>The TOE will provide a means to ensure users are not communicating with some other entity pretending to be the TOE, and that the TOE is communicating with an authorized IT entity and not some other entity pretending to be an authorized IT entity. | FTP_ITC.1(1), FTP_ITC.1(2)<br>FTP_TRP.1(1), FTP_TRP.1(2) | FTP_TRP.1.1 requires the TOE to provide a mechanism that creates a distinct communication path that protects the data that traverses this path from disclosure or modification. This requirement ensures that the TOE can identify the end points and ensures that a user cannot insert themselves between the user and the TOE, by requiring that the means used for invoking the communication path cannot be intercepted and allow a "man-in-the-middle-attack" (this does not prevent someone from capturing the traffic and replaying it at a later time – see FPT_RPL.1). Since the user invokes the trusted path (FTP_TRP.1.2) mechanism they can be assured they are communicating with the TOE. FTP_TRP.1.3 mandates that the trusted path be the only means available for providing identification and authentication information, therefore ensuring a user's authentication data will not be compromised when performing authentication functions. Furthermore, the remote administrator's communication path is encrypted during the entire session. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| | | FTP_ITC.1(1) and FTP_ITC.1(2) are similar to FTP_TRP.1(1) and FTP_TRP.1(2), in that they require a mechanism that creates a distinct communication path with the same characteristics, however FTP_ITC.1(1) and FTP_ITC.1(2) is used to protect communications between IT entities, rather than between a human user and an IT entity. FTP_ITC.1.3 requires the TOE to initiate the trusted channel, which ensures that the TOE has established a communication path with an authorized IT entity and not some other entity pretending to be an authorized IT entity. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| O.USER_GUIDANCE<br>The TOE will provide users with the information necessary to correctly use the security mechanisms. | AGD_USR.1 | O.USER_GUIDANCE (AGD_USR.1) mitigates this threat by providing the user the information necessary to use the security mechanisms that control access to user data in a secure manner. For instance, the method by which the discretionary access control mechanism (FDP_ACC.1, FDP_ACF.1) is configured, and how to apply it to the data the user owns, is described in the user guidance. If this information were not available to the user, the information may be left unprotected, or the user may mis-configure the controls and unintentionally allow unauthorized access to their data. |

| Objective | Requirements Addressing the Objective | Rationale |
|---|---|---|
| O.VULNERABILITY_ANALYSIS_TEST<br>The TOE will undergo appropriate independent vulnerability analysis and penetration testing to demonstrate the design and implementation of the TOE does not allow attackers with medium attack potential to violate the TOE's security policies. | AVA_VLA.3 | To maintain consistency with the overall assurance goals of this TOE, O.VULNERABILITY_ANALYSIS_TEST requires the AVA_VLA.3 component to provide the necessary level of confidence that vulnerabilities do not exist in the TOE that could cause the security policies to be violated. AVA_VLA.3 requires the developer to perform a systematic search for potential vulnerabilities in all the TOE deliverables. For those vulnerabilities that are not eliminated, a rationale must be provided that describes why these vulnerabilities cannot be exploited by a threat agent with a moderate attack potential, which is in keeping with the desired assurance level of this TOE. As with the functional testing, a key element in this component is that an independent assessment of the completeness of the developer's analysis is made, and more importantly, an independent vulnerability analysis coupled with testing of the TOE is performed. This component provides the confidence that security flaws do not exist in the TOE that could be exploited by a threat agent of moderate (or lower) attack potential to violate the TOE's security policies. |

## 6.4 Rationale for Assurance Requirements

191 The Evaluation Assurance Level (EAL) definitions and assurance requirements in Part 3 of the CC were used as a basis for the explicit assurance requirements developed by NSA for inclusion in Medium Robustness Protection Profile Assurance Requirements. Section 5.3 was believed to best achieve the goal of addressing circumstances where developers and users require a moderate level of independently assured security in commercial products. This collection of assurance requirements require TOE developers to gain assurance from high-quality software engineering development practices which, though rigorous, do not require substantial specialist knowledge, skills, and other resources. Rationale for individual assurance requirements is provided in Table 9 above. Rationale for explicit assurance requirements is provided in

Table 12 below.

## 6.5 Rationale for Strength of Function Claim

192 Part 1 of the CC defines “strength of function” in terms of the minimum efforts assumed necessary to defeat the expected security behaviour of a TOE security function. There are three strength of function levels defined in Part 1: SOF-basic, SOF-medium and SOF-high. SOF-medium is the strength of function level chosen for this PP. SOF-medium states, “a level of the TOE strength of function where analysis shows that the function provides adequate protection against straightforward or intentional breach of TOE security by attackers possessing a moderate attack potential.” The choice of SOF-medium is therefore consistent with the TOE objective O.VULNERABILITY_ANALYSIS_TEST and assurance requirements included in this PP. Specifically, AVA_VLA.3 requires that the TOE be resistant to an attacker with a moderate-attack potential, this is consistent with SOF-medium. Consequently, the metrics (i.e., passwords and keys) chosen for inclusion in this PP are acceptable for SOF-medium and would adequately protect information in a Medium Robustness Environment.

## 6.6 Rationale for Satisfying all Dependencies

**Table 11 Functional Requirement Dependencies**

| Requirement | Dependency | Satisfied |
|---|---|---|
| FCS_CKM.1<br>FCS_CKM.4 | FMT_MSA.2 | This dependency is satisfied by placing strict requirements on the values of attributes of the cryptographic module in the associated FCS requirements. Therefore, FMT_MSA.2 is not necessary to satisfy the requirement of only secure values being assigned to secure attributes. |
| FIA_UAU.1<br>FIA_UAU.2<br>FMT_SMR.2 | FIA_UID.1 | This dependency is satisfied with the inclusion of requirement FIA_UID.2. This requirement is hierarchical to FIA_UID.1 and is sufficient to satisfy the dependency for these requirements. |
| FMT_MOF.1<br>FMT_MSA.1<br>FMT_MTD.2<br>FMT_REV.1 | FMT_SMR.1 | This dependency is satisfied with the inclusion of requirement FMT_SMR.2. This requirement is hierarchical to FMT_SMR.1 and is sufficient to satisfy the dependency for these requirements. |

## 6.7 Rationale for Explicit Requirements

193

Table 12 presents the rationale for the inclusion of the explicit functional and assurance requirements found in this PP. The explicit requirements that are included as NIAP interpretations do not require a rationale for their inclusion per CCEVS management.

**Table 12 Rationale for Explicit Requirements**

| Explicit Requirement | Identifier | Rationale |
|---|---|---|
| FAU_ARP_ACK_(EXP).1 | Security alarm acknowledgement | This explicit requirement is necessary since a CC requirement does not exist to ensure an administrator will be aware of the alarm. The intent is to ensure that if an administrator is logged in and not physically at the console or remote workstation the message will remain displayed until the administrators have acknowledged it. The message will not be scrolled off the screen due to other activity-taking place (e.g., the auditor is running an audit report). |
| FCS_BCM_(EXP).1 | Baseline cryptographic module | This explicit requirement is necessary since the CC does not provide a means to specify a cryptographic baseline of implementation. |
| FCS_CKM_(EXP).1 | Cryptographic key validation and packing | This explicit requirement is necessary since the CC does not provide a means to specify a cryptographic key validation and packing techniques. |
| FCS_CKM_(EXP).2 | Cryptographic key validation and storage | This explicit requirement is necessary since the CC does not provide a means to specify a cryptographic key validation and storage method. |
| FCS_COA_(EXP).1 | Cryptographic operations availability | This explicit requirement is necessary since the CC does not provide a means to specify cryptographic operations (e.g. encryption, decryption, digital signature, etc.) |

| | | |
|---|---|---|
| FCS_COP_(EXP).1 | Random number generation | This explicit requirement is necessary since the CC does not provide a means to perform random number generation. This service is specified in NIST Special Publication 800-22. |
| FCS_IKE_(EXP).1 | Internet Key Exchange | This explicit requirement is necessary since the CC does not include requirements for this specific key enchange protocol. This protocol is specified in RFC 2409, but there are specific configurable setting that must be specified that are documented in the explicit requirement. |
| FIA_UAU_(EXP).5 | Multiple authentication mechanisms | This explicit requirement is needed for local administrators because there is no CC requirement that requires the TSF provide authentication. Because this PP allows the IT environment to provide an authentication server to be used for the single-use authentication mechanism for remote users, it is important to specify that the TSF provide the means for local administrator authentication in case the TOE cannot communicate with the authentication server. |

| | | |
|---|---|---|
| FPT_PRO_(EXP).1 | Standard protocol usage | This explicit requirement is necessary since the CC does not provide requirements of choosing a standard protocol mechanism from the standard protocols being used by a particular IT product. |
| FPT_TST_(EXP).4 | TSF testing (with cryptographic integrity verification) | This explicit requirement is necessary to capture the notion of the TOE using cryptography to verify the integrity of the TSF software. Additionally, the TSF data set that is subject to these tests was reduced to address the notion that it does not make sense to test the integrity of some TSF data (e.g., audit data) and this explicit requirement address that. |
| FPT_TST_(EXP).5 | Cryptographic self-test | The PP authors felt that the TSF self tests did not adequately address the notion of testing certain aspects of the TSF upon the completion of an operation. This explicit requirement is necessary to capture the notion of the TOE having the ability to test the cryptographic components immediately after the generation of a key. The CC does not contain a requirement that addresses this notion. |
| ADV_ARC_(EXP).1 | Architectural design with justification | These explicit assurance requirements were deemed necessary by NSA to red[illegible] |

| | | |
|---|---|---|
| ADV_FSP_(EXP).1 | Security-enforcing high-level design | necessary by NSA to reduce the ambiguity in the associated CC assurance families and to provide the level of assurance appropriate for medium robustness environments |
| ADV_HLD_(EXP).1 | Security-Enforcing High-Level Design | |
| ADV_INT_(EXP).1 | Modular decomposition | |
| ADV_LLD_(EXP).1 | Security-enforcing low-level design | |
| AVA_CCA_(EXP).2 | Systematic cryptographic module covert channel analysis | |

## 6.8 Rationale for Not Addressing Consistency Instructions

This Protection Profile conforms to the Medium Robustness Consistency Guidance except for the following instructions:

- Instruction 23 was not met because FDP_IFF.1-NIAP-0407 is no longer an active interp, CC V2.2 was used instead.
- Instruction 24 was not met because FIA-AFL.1-NIAP-0425 is no longer an active interp, CCIMB 111 was used instead.
- Instruction 25 was not met because FIA_USB.1-NIAP-0415 has been superseded by CCIMB 137.
- Instruction 27 was not met because FPT_RCV.2-NIAP-0406 is no longer an active interp, CC V2.2 was used instead.
- Instruction 32 was not met because FTA_TSE.1 does not need to be refined for this PP, CC V2.2 was used instead.
- This PP did not use the definition for O.MEDIATE because our main focus is protecting TSF data.

# 7 APPENDICES

194 Section 7 of this document contains the appendices, that accompany the PP and provides clarity and/or explanation for the reader.

# A REFERENCES


[1] Common Criteria for Information Technology Security Evaluation, *CCIMB-99-031, Version 2.2, January 2004.*

[2] U.S. Government Protection Profile for Single-Level Operating Systems in Environments Requiring Medium Robustness, *Version 1.67, 30 October 2003*

[3] *Department of Defense Chief Information Officer Guidance and Policy Memorandum No.6-8510*, Guidance and Policy for the Department of Defense Global Information Grid Information Assurance (GIG)*, June 2000.*

[4] U.S. Department of Defense Virtual Private Network (VPN) Boundary Gateway Protection Profile for Basic Robustness Environments (BRE*), Version .6, September 2001.*

[5] U.S. Government Firewall Protection Profile for Medium Robustness Environments, *Version 1.0, October 28, 2003*.

[6] Federal Information Processing Standard Publication (FIPS-PUB) 197, Advanced Encryption Standard (AES), November 2001.

[7] Federal Information Processing Standard Publication (FIPS-PUB) 140-2, Security Requirements for Cryptographic Modules, May 25, 2001.

[8] Federal Information Processing Standard (FIPS-PUB) 46-3, Data Encryption Standard (DES), October 1999.

[9] Internet Engineering Task Force, IP Encapsulating Security Payload (ESP), RFC 2406, November 1998.

[10] Internet Engineering Task Force, Internet Key Exchange (IKE), RFC 2409, November 1998.

[11] Information Assurance Technical Framework*, Release 3.0, September 2000*.

[12] Internet Engineering Task Force, ESP CBC-Mode Cipher Algorithms, RFC 2451, November 1998.

[13] Internet Engineering Task Force, Use of HMAC-SHA-1-96 within ESP and AH, RFC 2404, November 1998.

[14] Department of Defense Instruction, Information Assurance Implementation, 8500.2, February 6, 2003.

[15] Department of Defense Directive, Information Assurance, 8500.1, October 24, 2002.

[16] Implementing Virtual Private Networks, Steven Brown, McGraw Hill, 1999.

[17] A Goal VPN Protection Profile for Protecting Sensitive Information*, Release 2.0, July 2000.*

[18] The AES Cipher Algorithm and Its Use with Ipsec <draft-ietf-ipsec-ciph-aes-cbc.03.txt>, *Internet draft, November 2001*.

# B GLOSSARY

***Access*** – Interaction between an entity and an object that results in the flow or modification of data.

***Access Control*** – Security service that controls the use of resources[33] and the disclosure and modification of data.[34]

***Accountability*** – Property that allows activities in an IT system to be traced to the entity responsible for the activity.

***Administrator*** – A user who has been specifically granted the authority to manage some portion or the entire TOE and whose actions may affect the TSP. Administrators may possess special privileges that provide capabilities to override portions of the TSP.

***Assurance*** – A measure of confidence that the security features of an IT system are sufficient to enforce its' security policy.

***Asymmetric Cryptographic System*** – A system involving two related transformations; one determined by a public key (the public transformation), and another determined by a private key (the private transformation) with the property that it is computationally infeasible to determine the private transformation (or the private key) from knowledge of the public transformation (and the public key).

***Asymmetric Key*** – The corresponding public/private key pair needed to determine the behavior of the public/private transformations that comprise an asymmetric cryptographic system

***Attack*** – An intentional act attempting to violate the security policy of an IT system.

***Authentication*** – Security measure that verifies a claimed identity.

***Authentication data*** – Information used to verify a claimed identity.

***Authorization*** – Permission, granted by an entity authorized to do so, to perform functions and access data.

[33] Hardware and software.
[34] Stored or communicated.

***Authorized user*** – An authenticated user who may, in accordance with the TSP, perform an operation.

***Availability*** – Timely[35], reliable access to IT resources.

***Compromise*** – Violation of a security policy.

***Confidentiality*** – A security policy pertaining to disclosure of data.

***Critical Security Parameters (CSP)*** – Security-related information (e.g., cryptographic keys, authentication data such as passwords and pins, and cryptographic seeds) appearing in plaintext or otherwise unprotected form and whose disclosure or modification can compromise the security of a cryptographic module or the security of the information protected by the module.

***Cryptographic Administrator*** – An authorized user who has been granted the authority to perform cryptographic initialization and management functions. These users are expected to use this authority only in the manner prescribed by the guidance given to them.

***Cryptographic boundary*** – An explicitly defined contiguous perimeter that establishes the physical bounds (for hardware) or logical bounds (for software) of a cryptographic module.

***Cryptographic key (key)*** – A parameter used in conjunction with a cryptographic algorithm that determines:

- the transformation of plaintext data into ciphertext data,
- the transformation of ciphertext data into plaintext data,
- a digital signature computed from data,
- the verification of a digital signature computed from data, or
- a digital authentication code computed from data.

[35] According to a defined metric.

***Cryptographic Module*** – The set of hardware, software, firmware, or some combination thereof that implements cryptographic logic or processes, including cryptographic algorithms, and is contained within the cryptographic boundary of the module.

***Cryptographic Module Security Policy*** – A precise specification of the security rules under which a cryptographic module must operate, including the rules derived from the requirements of this PP and additional rules imposed by the vendor.

***Defense-in-Depth (DID)*** – A security design strategy whereby layers of protection are utilized to establish an adequate security posture for an IT system.

***Discretionary Access Control (DAC)*** – A means of restricting access to objects based on the identity of subjects and/or groups to which they belong. Those controls are discretionary in the sense that a subject with certain access permission is capable of passing that permission (perhaps indirectly) on to any other subject.

***Embedded Cryptographic Module*** – On that is built as an integral part of a larger and more general surrounding system (i.e., one that is not easily removable from the surrounding system).

***Enclave*** – A collection of entities under the control of a single authority and having a homogeneous security policy. They may be logical, or may be based on physical location and proximity.

***Entity*** – A subject, object, user or another IT device, which interacts with TOE objects, data, or resources.

***External IT entity*** – Any trusted Information Technology (IT) product or system, outside of the TOE, which may, in accordance with the TSP, perform an operation.

***Identity*** – A representation (e.g., a string) uniquely identifying an authorized user, which can either be the full or abbreviated name of that user or a pseudonym.

***Integrity*** – A security policy pertaining to the corruption of data and TSF mechanisms.

***Integrity label*** – A security attribute that represents the integrity level of a subject or an object. Integrity labels are used by the OTE as the basis for mandatory integrity control decisions.

***Integrity level*** – The combination of a hierarchical level and an optional set of non-hierarchical categories that represent the integrity of data.

***Mandatory Access Control (MAC)*** – A means of restricting access to objects based on subject and object sensitivity labels.[36]

***Mandatory Integrity Control (MIC)*** – A means of restricting access to objects based on subject and object integrity labels.

***Multilevel*** – The ability to simultaneously handle (e.g., share, process) multiple levels of data, while allowing users at different sensitivity levels to access the system concurrently. The system permits each user to access only the data to which they are authorized access.

***Named Object*** – An object that exhibits all of the following characteristics:

- The object may be used to transfer information between subjects of differing user identities within the TSF.
- Subjects in the TOE must be able to require a specific instance of the object.
- The name used to refer to a specific instance of the object must exist in a context that potentially allows subjects with different user identities to require the same instance of the object.

***Non-Repudiation*** – A security policy pertaining to providing one or more of the following:

- To the sender of data, proof of delivery to the intended recipient,
- To the recipient of data, proof of the identity of the user who sent the data.

***Object*** – An entity within the TSC that contains or receives information and upon which subjects perform operations.

***Operating Environment*** – The total environment in which a TOE operates. It includes the physical facility and any physical, procedural, administrative and personnel controls.

[36] The Bell LaPadula model is an example of Mandatory Access Control.

***Operating System (OS)*** – An entity within the TSC that causes operations to be performed. Subjects can come in two forms: trusted and untrusted. Trusted subjects are exempt from part or all of the TOE security policies. Untrusted subjects are bound by all TOE security policies.

***Operational key*** – Key intended for protection of operational information or for the production or secure electrical transmissions of key streams

***Peer TOEs*** – Mutually authenticated TOEs that interact to enforce a common security policy.

***Public Object*** – An object for which the TSF unconditionally permits all entities "read" access. Only the TSF or authorized administrators may create, delete, or modify the public objects.

***Robustness*** – A characterization of the strength of a security function, mechanism, service or solution, and the assurance (or confidence) that it is implemented and functioning correctly. DoD has three levels of robustness:

***Basic:*** Security services and mechanisms that equate to good commercial practices.

***Medium:*** Security services and mechanisms that provide for layering of additional safeguards above good commercial practices.

***High:*** Security services and mechanisms that provide the most stringent protection and rigorous security countermeasures.

***Secure State*** – Condition in which all TOE security policies are enforced.

***Security attributes*** – TSF data associated with subjects, objects, and users that are used for the enforcement of the TSP.

***Security level*** – The combination of a hierarchical classification and a set of non-hierarchical categories that represent the sensitivity of the information.

***Sensitivity label*** – A security attribute that represents the security level of an object and that describes the sensitivity (e.g., Classification) of the data in the object. Sensitivity labels are used by the TOE as the basis for mandatory access control decision.

***Split key*** – A variable that consists of two or more components that must be combined to form the operation key variable. The combining process excludes concatenation or interleaving of component variables.

***Subject*** – An entity within the TSC that causes operation to be performed.

***Symmetric key*** – A single, secret key used for both encryption and decryption in symmetric cryptographic algorithms.

***Threat*** – Capabilities, intentions and attack methods of adversaries, or any circumstance or event, with the potential to violate the TOE security policy.

***Threat Agent*** – Any human user or Information Technology (IT) product or system, which may attempt to violate the TSP and perform an unauthorized operation with the TOE.

***User*** – Any entity (human user or external IT entity) outside the TOE that interacts with the TOE.

***Vulnerability*** – A weakness that can be exploited to violate the TOE security policy.

# C ACRONYMS

| | |
|---|---|
| ANSI | American National Standards Institute |
| BGP | Border Gateway Protocol |
| CC | Common Criteria |
| CCEVS | Common Criteria Evaluation Validation Scheme |
| CCIMB | Common Criteria Interpretations Management Board |
| CCM | Counter with Cipher Block Chaining-Message Authentication Code |
| CLNP | Connectionless Network Protocol |
| CLNP | Connectionless Network Protocol |
| CLNS | Connectionless Network Service |
| CM | Configuration Management |
| CSP | Cryptographic security parameter |
| DoD | Department of Defense |
| EAL | Evaluation Assurance Level |
| FIPS | Federal Information Processing Standard |
| FTP | File Transfer Protocol |
| GIG | Global Information Grid |
| HMAC | Keyed-Hash Authentication Code |
| HTTP | Hypertext Transfer Protocol |
| IATF | Information Assurance Technical Framework |
| ICMP | Internet Control Message Protocol |

| | |
|---|---|
| IKE | Internet Key Exchange |
| IPSEC | Internet Protocol Security |
| IPv6 | Internet Protocol Version 6 |
| IPX | Internetwork Packet Exchange |
| IPX | Internetwork Packet Exchange |
| ISAKMP | Internet Security Association and Key Management Protocol |
| IS-IS | Intermediate System-to-Intermediate System |
| ISO | International Organization for Standardization |
| IT | Information Technology |
| LDP | Label Distribution Protocol |
| MAC | Mandatory Access Control |
| MPLS | Multi-protocol Label Switching |
| NIAP | National Information Assurance Partnership |
| NIST | National Institute of Standards Technology |
| NSA | National Security Agency |
| NTP | Network Time Protocol |
| OSI | Open Systems Interconnect |
| OSPF | Open Shortest Path First |
| PKI | Public Key Infrastructure |
| PP | Protection Profile |
| PRNG | Prime Random Number Generator |
| RFC | Request for Comments |

| | |
|---|---|
| RIP | Routing Information Protocol |
| RNG | Random Number Generator |
| RSA | Rivest, Shamir, Adelman |
| SA | Security Association |
| SFP | Security Functional Policy |
| SFR | Security Functional Requirement |
| SHA | Secure Hash Algorithm |
| SOF | Strength of Function |
| SOF | Strength of Function |
| ST | Security Target |
| TBD | To Be Determined |
| TCP/IP | Transmissions Control Protocol/ Internet Protocol |
| TDEA | Triple Data Encryption Algorithm |
| TFTP | Trivial File Transfer Protocol |
| TOE | Target of Evaluation |
| TSC | TOE Scope of Control |
| TSE | TOE Security Environment |
| TSF | TOE Security Functions |
| TSFI | TSF interfaces |
| TSP | TOE Security Policy |
| TTAP/CCEVS | Trust Technology Assessment Program/ Common Criteria Evaluation Standard Scheme |
| UDP | User Datagram Protocol |

| | |
|---|---|
| VPN | Virtual Private Network |

# D ROBUSTNESS ENVIRONMENT CHARACTERIZATION

## D.1 General Environmental Characterization

195 In trying to specify the environments in which TOEs with various levels of robustness are appropriate, it is useful to first discuss the two defining factors that characterize that environment: value of the resources and authorization of the entities to those resources.

196 In general terms, the environment for a TOE can be characterized by the authorization (or lack of authorization) the least trustworthy entity has with respect to the highest value of TOE resources (i.e., the TOE itself and all of the data processed by the TOE).

197 Note that there are an infinite number of combinations of entity authorization and value of resources; this conceptually "makes sense" because there are an infinite number of potential environments, depending on how the resources are valued by the organization, and the variety of authorizations the organization defines for the associated entities. In the next section, these two environmental factors will be related to the robustness required for selection of an appropriate TOE.

### D.1.1 Value of Resources

198 Value of the resources associated with the TOE includes the data being processed or used by the TOE, and the TOE itself (for example, a real-time control processor). "Value" is assigned by the using organization. For example, in the DoD low-value data might be equivalent to data marked "FOUO", while high-value data may be those classified Top Secret. In a commercial enterprise, low-value data might be the internal organizational structure as captured in the corporate on-line phone book, while high-value data might be corporate research results for the next generation product. Note that when considering the value of the data one must also consider the value of data or resources that are accessible through exploitation of the TOE. For example, a firewall may have "low value" data itself, but it might protect an enclave with high value data. If the firewall was being depended upon to protect the high value data, then it must be treated as a high-value-data TOE.

### D.1.2 Authorization of Entities

199 Authorization that entities (users, administrators, and other IT systems) have with respect to the TOE (and thus the resources of that TOE, including the TOE itself) is an abstract concept reflecting a combination of the trustworthiness of an entity and the access and privileges granted to that entity with respect to the resources of the TOE. For instance, entities that have total authorization to all data on the TOE are at one end of this spectrum; these entities may have privileges that allow them to read, write, and modify anything on the TOE, including all TSF data. Entities at the other end of the spectrum are those that are authorized to few or no TOE resources. For example, in the case of a router, non-administrative entities may have their packets routed by the TOE, but that is the extent of their authorization to the TOE's resources. In the case of an OS, an entity may not be allowed to log on to the TOE at all (that is, they are not valid users listed in the OS's user database).

200 It is important to note that authorization **does not** refer to the **access** that the entities actually have to the TOE or its data. For example, suppose the owner of the system determines that no one other than employees was authorized to certain data on a TOE, yet they connect the TOE to the Internet. There are millions of entities that are not **authorized** to the data (because they are not employees), but they actually have connectivity to the TOE through the Internet and thus can attempt to access the TOE and its associated resources.

201 Entities are characterized according to the value of resources to which they are authorized; the extent of their authorization is implicitly a measure of how trustworthy the entity is with respect to compromise of the data (that is, compromise of any of the applicable security policies; e.g., confidentiality, integrity, availability). In other words, in this model the greater the extent of an entity's authorization, the more trustworthy (with respect to applicable policies) that entity is.

### D.1.3 Selection of Appropriate Robustness Levels

202 Robustness is a characteristic of a TOE defining how well it can protect itself and its resources; a more robust TOE is better able to protect itself. This section relates the defining factors of IT environments, authorization, and value of resources to the selection of appropriate robustness levels.

203 When assessing any environment with respect to Information Assurance the critical point to consider is the likelihood of an attempted security policy compromise, which was characterized in the previous section in terms of entity authorization and resource value. As previously mentioned, robustness is a characteristic of a TOE that reflects the extent to which a TOE can protect itself

and its resources. It follows that as the likelihood of an attempted resource compromise increases, the robustness of an appropriate TOE should also increase.

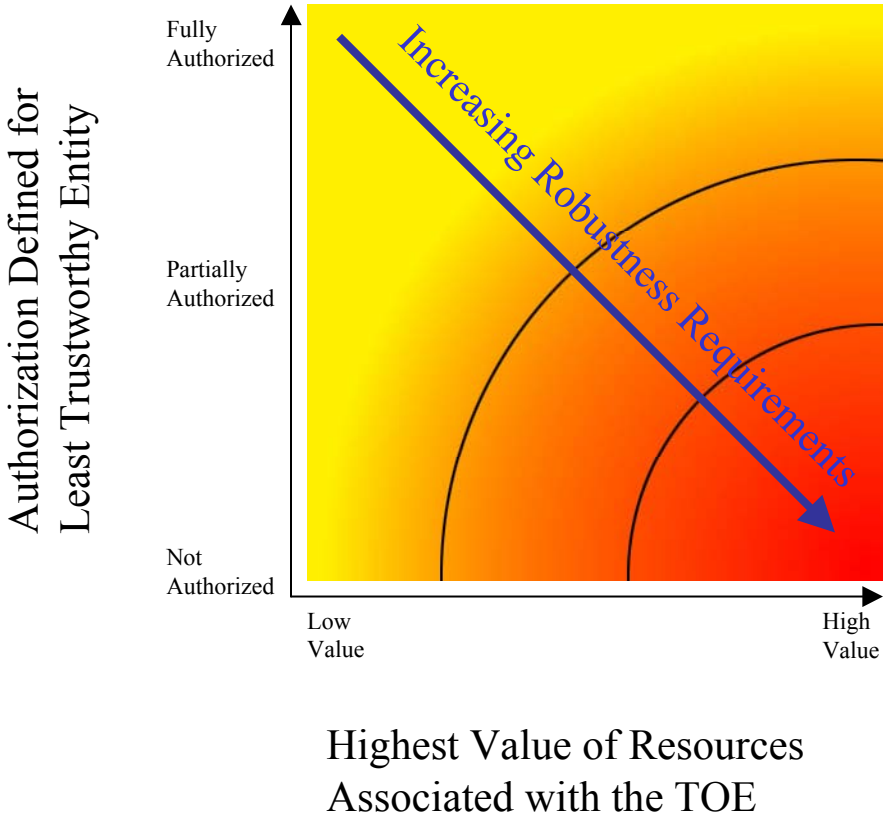
204 It is critical to note that several combinations of the environmental factors will result in environments in which the likelihood of an attempted security policy compromise is similar. Consider the following two cases:

205 The first case is a TOE that processes only low-value data. Although the organization has stated that only its employees are authorized to log on to the system and access the data, the system is connected to the Internet to allow authorized employees to access the system from home. In this case, the least trusted entities would be unauthorized entities (e.g. non-employees) exposed to the TOE because of the Internet connectivity. However, since only low-value data are being processed, the likelihood that unauthorized entities would find it worth their while to attempt to compromise the data on the system is low and selection of a basic robustness TOE would be appropriate.

206 The second case is a TOE that processes high-value (e.g., classified) information. The organization requires that the TOE be stand-alone, and that every user with physical and logical access to the TOE undergo an investigation so that they are authorized to the highest value data on the TOE. Because of the extensive checks done during this investigation, the organization is assured that only highly trusted users are authorized to use the TOE. In this case, even though high value information is being processed, it is unlikely that a compromise of that data will be attempted because of the authorization and trustworthiness of the users and once again, selection of a basic robustness TOE would be appropriate.

207 The preceding examples demonstrated that it is possible for radically different combinations of entity authorization/resource values to result in a similar likelihood of an attempted compromise. As mentioned earlier, the robustness of a system is an indication of the protection being provided to counter compromise attempts. Therefore, a basic robustness system should be sufficient to counter compromise attempts where the likelihood of an attempted compromise is low. The following chart depicts the "universe" of environments characterized by the two factors discussed in the previous section: on one axis is the authorization defined for the least trustworthy entity, and on the other axis is the highest value of resources associated with the TOE.

208 As depicted in the following figure, the robustness of the TOEs required in each environment steadily increases as one goes from the upper left of the chart to the lower right; this corresponds to the need to counter increasingly likely attack attempts by the least trustworthy entities in the environment. Note that the shading




of the chart is intended to reflect- the notion that different environments engender similar levels of "likelihood of attempted compromise", signified by a similar color. Further, the delineations between such environments are not stark, but rather are finely grained and gradual.

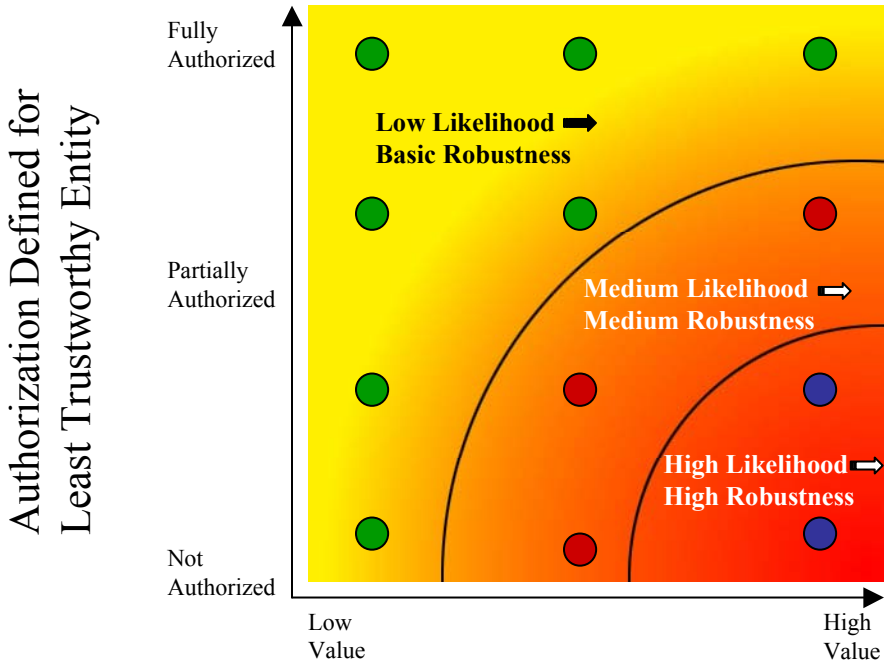
209 While it would be possible to create many different "levels of robustness" at small intervals along the "Increasing Robustness Requirements" line to counter the increasing likelihood of attempted compromise due to those attacks, it would not be practical neither particularly useful. Instead, in order to implement the robustness strategy where there are only three robustness levels: Basic, Medium, and High, the graph is divided into three sections, with each section corresponding to a set of environments where the likelihood of attempted compromise is roughly similar. This is graphically depicted in the following chart.

210 In this second representation of environments and the robustness plane below, the "dots" represent given instantiations of environments; like-colored dots define environments with a similar likelihood of attempted compromise. Correspondingly, a TOE with a given robustness should provide sufficient

protection for environments characterized by like-colored dots. In choosing the appropriateness of a given robustness level TOE PP for an environment, then, the user must first consider the lowest authorization for an entity and the highest value of the resources in that environment. This should result in a "point" in the chart above, corresponding to the likelihood that that entity will attempt to compromise the most valuable resource in the environment. The appropriate robustness level for the specified TOE to counter this likelihood can then be chosen.

211 The difficult part of this activity is differentiating the authorization of various entities, and determining the relative values of resources; (e.g., what constitutes "low value" data vs. "medium value" data). Because every organization will be different, a rigorous definition is not possible. In Section 3 of this PP, the targeted threat level for a medium robustness TOE is characterized. This information is provided to help organizations using this PP -ensure that the functional requirements specified by this medium robustness PP are appropriate for their intended application of a compliant TOE.

**E**



# EXPLANATORY MATERIAL FOR EXPLICIT ASSURANCE REQUIREMENTS

## E.1 ADV_INT_(EXP).1

212 This explicit component was created to levy different modularity metrics on the SFP-enforcing modules and non-SFP-enforcing modules.

213 The parts of the TSF that implement an SFP (in this component, SFP-enforcing is used to designate modules that enforce an SFP) that is determined and assigned by the PP/ST author, are those modules that interact (defined in the coupling analysis) with the module or modules that provide the TSFI for that SFP with justified exceptions. The intent is that all of the modules that play an SFP related role (as opposed to modules that provide infrastructure support, such as scheduling, reading binary data from the disk) in enforcing an SFP are identified as SFP-enforcing. The remaining modules in the TSF are deemed non-SFP-enforcing modules, since they could be TSP-enforcing (e.g., enforcing a policy not assigned to this component), and TSP-supporting.

### E.1.1 Objectives

214 This component addresses the internal structure of the software TSF. The SFP-enforcing modules require stricter adherence to the coupling and cohesion metrics than the metrics levied on the non-SFP-enforcing modules due to their key role in policy enforcement. While the non-SFP-enforcing modules also play a role in enforcing policy, their role is not as critical as the SFP-enforcing modules; therefore, the degree of coupling and cohesion required of these modules is not as restrictive. It is expected that all of the TSF modules are designed using high-quality software engineering practice, whether they are developed by the developer or incorporated as a third party implementation into the TSF.

215 Requirements are presented for modular decomposition of the SFP-enforcing and non-SFP-enforcing functionality within the TSF. These requirements, when applied to the internal structure of the TSF, should result in improvements that aid both the developer and the evaluator in understanding the TSF, and also provides the basis for designing and evaluating test suites. Further, improving understandability of the TSF should assist the developer in simplifying its maintainability. The principal goal achieved by inclusion of the requirements from the ADV_INT class in a PP/ST is understandability of the TSF.

216 Modular design aids in achieving understandability by clarifying what dependencies and interactions a module has on other modules (*coupling*), by including in a module only tasks that are strongly related to each other (*cohesion*), and by illuminating the design of a module by using internal structuring and reduced complexity. The use of modular design reduces the interdependence between elements of the TSF and thus reduces the risk that a change or error in one module will have effects throughout the TOE. Its use enhances clarity of design and provides for increased assurance that unexpected effects do not occur. Additional desirable properties of modular decomposition are a reduction in the amount of redundant or unneeded code.

217 The incorporation of modular decomposition into the design and implementation process must be accompanied by sound software engineering considerations. A practical, useful software system will usually entail some undesirable coupling among modules, some modules that include loosely-related functions, and some subtlety or complexity in a module's design. These deviations from the ideals of modular decomposition are often deemed necessary to achieve some goal or constraint, be it related to performance, compatibility, future planned functionality, or some other factors, and may be acceptable, based on the developer's justification for them. In applying the requirements of this class, due consideration must be given to sound software engineering principles; however, the overall objective of achieving understandability must be achieved.

218 Another key component to reducing complexity is the use of coding standards. Coding standards are used as a reference to ensure programmers generate code that can be easily understood by individuals (e.g., code maintainers, code reviewers, evaluators) that are not intimately familiar with the nuances of the functions performed by the code. For example, coding standards ensure that meaningful names are given to variables and data structures, the code has a structure that is similar to code developed by other programmers, loops used in the code are understandable (e.g., leaving a loop to another section of code and returning is undesirable), the use of pointers to variables/data structures is straightforward, and the code is suitably commented (inline and/or by a preamble). The use of coding standards helps to eliminate errors in code development and maintenance, and assists the development team in performing code walk-throughs. Some aspects of coding standards are specific to a given program language (e.g., the C language may have a different standard than the Java language or assembly level code). It is expected that the coding standards are appropriately followed for the employed programming language(s). The requirements in this component allow for exceptions to the adherence of coding standards that may be necessary for reasons of performance, or some other factors, but these deviations must be justified (on a per module basis) as to why they are necessary. Any justification provided must

address why the deviation does not unduly introduce complexity into the module, since ultimately, the goal of adhering to coding standards is to improve clarity.

219 Design complexity minimization is a key characteristic of a reference validation mechanism, the purpose of which is to arrive at a TSF that is easily understood so that it can be completely analyzed. (There are other important characteristics of a reference validation mechanism, such as TSF self-protection and TSP non-bypassability; these other characteristics are covered by requirements from other classes.)

### E.1.2 Application Notes

220 Several of the elements within this component refer to the architectural description. The architectural description is at a similar level of abstraction as the low-level design, in that it is concerned with the modules of the TSF. Whereas the low-level design describes the design of the modules of the TSF, the purpose of the architectural description is to provide evidence of modular decomposition of the TSF. Both the low-level design and the implementation representation are required to be in compliance with the architectural description, to provide assurance that these TSF representations possess the required modular decomposition.

221 This component requires the PP or ST author to fill in an assignment with the SFPs that are felt to be critical to the TOE and therefore their resulting design and implementation require stricter metrics for modularity. The SFPs can be those explicitly identified in the CC (i.e., FDP_ACC, FDP_IFF) by simply placing the appropriate label as specified in those requirements, or other policies determined by the PP/ST author (e.g., I&A, Audit), in which case, the PP/ST author should explicitly identify all of the SFRs that they intend to satisfy a policy that is not explicitly stated in the CC. This is necessary since currently a convention does not exist to place a convenient label on these policies.

222 The requirements in this component refer to SFP-enforcing and non-SFP-enforcing

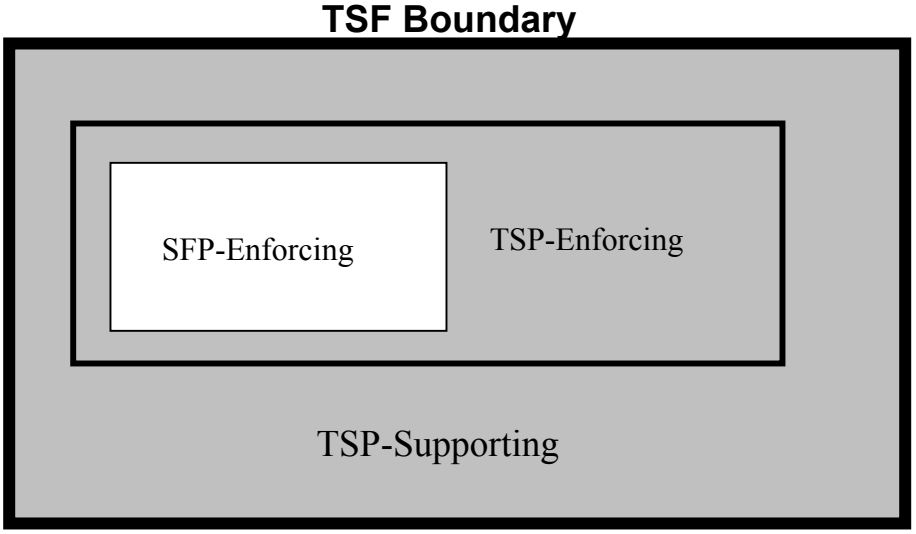



**Figure E1. SFP-enforcing may only be a subset of TSP-enforcing functions.**

portions of the TSF. The non-SFP-enforcing portions of the TSF consist of the TSP-supporting modules and TSP-enforcing modules that do not play a role in the enforcement of the SFP(s) identified in ADV_INT_(EXP).1.4D as depicted in the Figure E1, where in this example, non-SFP-enforcing is everything in the TSF other than the SFP-enforcing functions.

223 The developer is required to identify the modules that are SFP-enforcing and implicitly the remaining modules, which will be non-SFP-enforcing. As stated earlier, the SFP-enforcing modules are those modules that interact with the module or modules that provide the TSFI for that SFP with justified exceptions. The justification of the non-SFP-enforcing modules (ADV_INT_(EXP).1.3C) is required only for those modules that interact with SFP-enforcing modules and not for all non-SFP-enforcing modules. As depicted in the Figure E2 below, if a TSFI has already been designated as non-SFP-enforcing then the designation of the modules interacting with the module providing the TSFI do not have to be justified (e.g., modules X, Y, Z). The justification of the designation is only necessary for the module(s) that interact with a module that provides a TSFI that is SFP-enforcing (e.g., modules D, E, F (since it is writing to a global variable that Module A is reading, but in this example, it is not an SFP-enforcing variable).

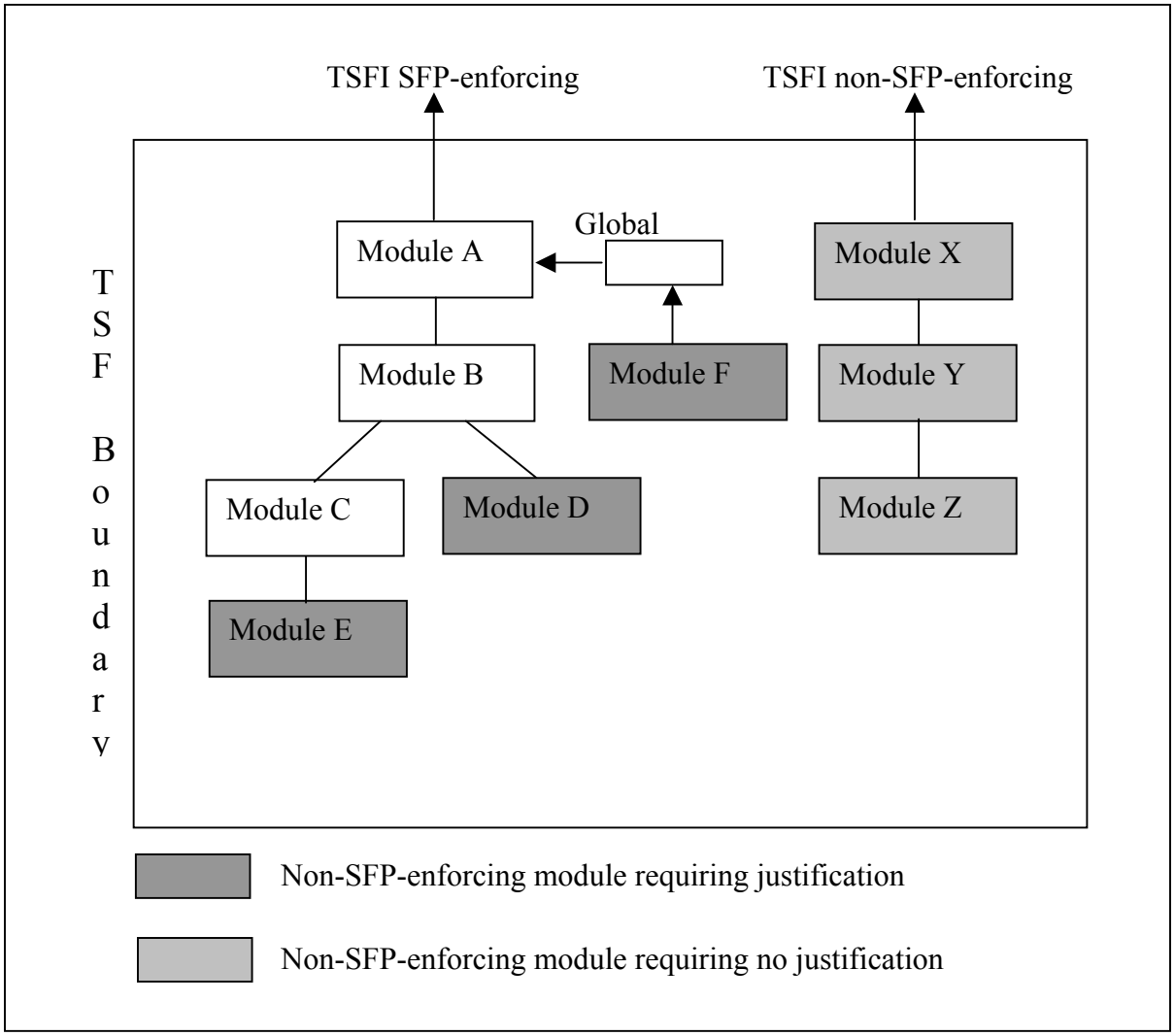



**Figure E2. Example of non-SFP-enforcing modules requiring justification.**

224 The modules identified in the architectural description are the same as the modules identified in the low-level design.

### E.1.3 Terms, Definitions and Background

225 The following terms are used in the requirements for software internal structuring. Some of these are derived from the Institute of Electrical and Electronics Engineers *Glossary of software engineering terminology, IEEE Std 610.12-1990*.

***Module*** – One or more source code files that cannot be decomposed into smaller compliable units.

***Modular decomposition*** – The process of breaking a system into components to facilitate design and development.

***Cohesion (also called module strength)*** – The manner and degree to which the tasks are performed by a single software module are related to one another; types of cohesion include coincidental, communicational, functional, logical, sequential, and temporal. These types of cohesion are characterized below, listed in order of decreasing desirability.

***Functional cohesion*** – A module with this characteristic performs activities related to a single purpose. A functionally cohesive module transforms a single type of input into a single type of output, such as a stack manager or a queue manager.

***Sequential cohesion*** – A module with this characteristic contains functions each of whose output is input for the following function in the module. An example of a sequentially cohesive module is one that contains the functions to write audit records and to maintain a running count of the accumulated number of audit violations of a specified type.

***Communicational cohesion*** – A module with this characteristic contains functions that produce output for, or use output from, other functions within the module. An example of a communicationally cohesive module is an access check module that includes mandatory, discretionary, and capability checks.

***Temporal cohesion*** – A module with this characteristic contains functions that need to be executed at about the same time. Examples of temporally cohesive modules include initialization, recover, and shutdown modules.

***Logical (or procedural) cohesion*** – A module with this characteristic performs similar activities on different data structures. A module exhibits logical cohesion if its functions perform related, but different, operations on different inputs.

***Coincidental cohesion*** – A module with this characteristic performs unrelated or loosely related activities.

***Coupling*** – The manner and degree of interdependence between software modules; types of coupling include call, common and content coupling. These types of coupling are characterized below, listed in the order of decreasing desirability.

***Call*** – Two modules are call coupled if they communicate strictly through the use of their documented function calls; examples of call coupling are data, stamp, and control, which are defined below.

***Data*** – Two modules are data coupled if they communicate strictly through the use of call parameters that represent single data items.

***Stamp*** – Two modules are stamp coupled if they communicate through the use of call parameters that comprise multiple fields or that have meaningful internal structures.

***Control*** – Two modules are control coupled if one passes information that is intended to influence the internal logic of the other.

***Common*** – Two modules are common coupled if they share a common data area or a common system resource. Global variables indicate that modules using those global variables are common coupled.[37]

Common coupling through global variables is generally allowed, but only to a limited degree. For example, variables that are placed into a global area, but are used by only a single module, are inappropriately placed, and should be removed. Other factors that need to be considered in assessing the suitability of global variables are:

[37] It can be argued that modules sharing definitions, such as data structure definitions, are common coupled. However, for the purposes of this analysis, shared definitions are considered acceptable, but are subject to the cohesion analysis.

The number of modules that modify a global variable: In general, only a single module should be allocated the responsibility for controlling the contents of a global variable, but there may be situations in which a second module may share that responsibility; in such a case, sufficient justification must be provided. It is unacceptable for this responsibility to be shared by more than two modules. (In making this assessment, care should be given to determining the module actually responsible for the contents of the variable; for example, if a single routine is used to modify the variable, but that routine simply performs the modification requested by its caller, it is the calling module that is responsible, and there may be more than one such module). Further, as part of the complexity determination, if two modules are responsible for the contents of a global variable, there should be clear indications of how the modifications are coordinated between them.

The number of modules that reference a global variable: Although there is generally no limit on the number of modules that reference global variable, cases in which many modules make such a reference should be examined for validity and necessity.

***Content*** – Two modules are content coupled if one can make direct reference to the internals of the other (e.g., modifying code of, or referencing labels internal to, the other module). The result is that some or all of the content of one module are effectively included in the other. Content coupling can be though of as using unadvertised module interfaces; this is in contract to call coupling, which uses only advertised module interfaces.

***Call tree*** – A diagram that identifies the modules in a system and shows which modules call one another. All the modules named in a call tree that originates with (i.e., is rooted by) a specific module are the modules that directly or indirectly implement the functions of the originating module.

***Software engineering*** - The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software. As with engineering practices in general, some amount of judgment must be used in applying engineering principles. Many factors affect choices, not just the application of measures of modular decomposition, layering, and minimization. For example, a developer may design a system with future applications in mind that will not be implemented initially. The developer may choose to include some logic to handle these future applications without fully implementing them; further, the developer may include some calls to as-yet unimplemented modules, leaving *call stubs*. The developer's

justification for such deviations from well-structured programs will have to be assessed using judgment, and the application of good software engineering discipline.

***Complexity*** - This is a measure of how difficult software is to understand, and thus to analyze, test, and maintain. Reducing complexity is the ultimate goal for using modular decomposition, layering and minimization. Controlling coupling and cohesion contributes significantly to this goal.

226 A good deal of effort in the software engineering field has been expended in attempting to develop metrics to measure the complexity of source code. Most of these metrics use easily computed properties of the source code, such as the number of operators and operands, the complexity of the control flow graph (*cyclomatic complexity*), the number of lines of source code, the ratio of comments to executable code, and similar measures. Coding standards have been found to be a useful tool in generating code that is more readily understood.

227 While this component calls for the evaluator to perform a *complexity analysis*, it is expected that the developer will provide support for the claims that the modules are not overly complex (ADV_INT_(EXP).1.3D, ADV_INT_(EXP).1.6D, ADV_INT_(EXP).1.9C). This support could include the developer's programming standards, and an indication that all modules meet the standard (or that there are some exceptions that are justified by software engineering arguments). It could include the results of tools used to measure some of the properties of the source code. Or it could include other support that the developer finds appropriate.

## E.2 ADV_FSP_(EXP).1

228 The functional specification is a description of the user-visible interface to the TSF. It contains an instantiation of the TOE security functional requirements. The functional specification has to completely address all of the user-visible TOE security functional requirements.

### E.2.1 Application Notes

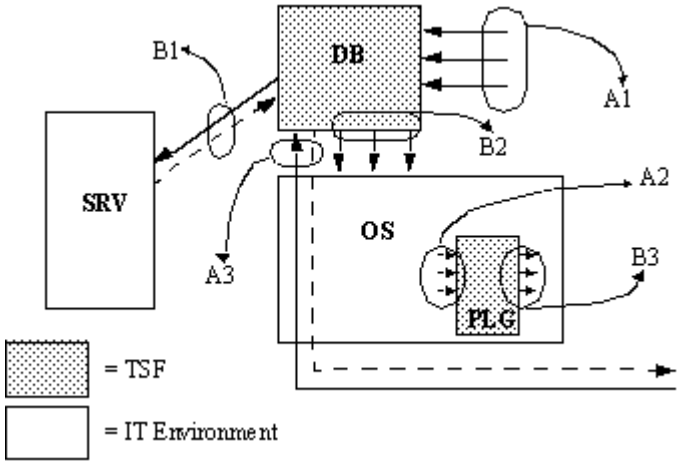
229 A description of the TSF interfaces (TSFI) provides fundamental evidence on which assurance in the TOE can be built. Fundamentally, the functional specification provides a description of *what* the TSF provides to users (as opposed to the high-level design and low-level design, which provide a description of *how* the functionality is provided). Further, the functional specification provides this information in the form of interface (TSFI) documentation.

230 In order to identify the software interfaces to the TSF, the parts of the TOE that make up the TSF must be identified. This identification is formally a part of ADV_HLD_EXP analysis. In this analysis, a portion of the TOE is considered to be in the TSF under two conditions:

1. The software contributes to the satisfaction of security functionality specified by a functional requirement in the ST. This is typically all software that runs in a privileged state of the underlying hardware, as well as software that runs in unprivileged states that performs security functionality.

2. The software used by administrators in order to perform security management activities specified in the guidance documentation. These activities are a superset of those specified by any FMT_* functional requirements in the ST.

231 Identification of the TSFI is a complex undertaking. The TSF is providing services and resources, and so the TSFI are interfaces *to* the security services/resources the TSF is providing. This is especially relevant for TSFs that have dependencies on the IT environment, because not only is the TSF providing security services (and thus exposing TSFI), but it is also *using* services of the IT environment. While these are (using the general term) interfaces between the TSF and the IT environment, they are not TSFI. Nonetheless, it is vital to document their existence to integrators and consumers of the system, and thus documentation requirements for these interfaces are specified in ADV_INT.




232 This concept (and concepts to be discussed in the following paragraphs) is illustrated in the following figure.

233 The figure above illustrates a TOE (a database management system) that has dependencies on the IT environment. The shaded boxes represent the TSF, while the un-shaded boxes represent IT entities in the environment. The TSF comprises the database engine and management GUIs (represented by the box labeled "DB") and a kernel module that runs as part of the OS that performs some security function (represented by the box labeled "PLG"). The TSF kernel module has entry points defined by the OS specification that the OS will call to invoke some function (this could be a device driver, or an authentication module, etc.). The key is that this pluggable kernel module is providing security services specified by functional requirements in the ST. The IT environment consists of the operating system (represented by the box labeled "OS") itself, and an external server (labeled SRV). This external server, like the OS, provides a service that the TSF depends on, and thus needs to be in the IT environment. Interfaces in the figure are labeled Ax for TSFI, and Bx for interfaces to be documented in ADV_INT. Each of these groups of interfaces is now discussed.

234 Interface group A1 represents the prototypical set of TSFI. These are interfaces used to directly access the database and its security functionality and resources.

235 Interface group A2 represent the TSFI that the OS invokes to obtain the functionality provided by the pluggable module. These are contrasted with interface group B3, which represent calls that the pluggable module makes to obtain services from the IT environment.

236 Interface group A3 represents TSFI that "pass through" the IT environment. In this case, the DBMS communicates over the network using a proprietary

application-level protocol. While the IT environment is responsible for providing various supporting protocols (e.g., Ethernet, IP, TCP), the application layer protocol that is used to obtain services from the DBMS is a TSFI and must be documented as such. The dotted line indicates return values/services from the TSF over the network connection.

237 Non-TSFI interfaces pictured are labeled Bx. Interface group B1 is the most complex of these, because the architecture of the system and environmental assumptions and conditions will drive its analysis. In the first case, assume that, either through an environmental assumption or an IT environmental requirement, the network link between the DB and SRV is protected (it could be on a separate subnet, or it could be protected by a firewall such that only the DB could connect to the port on the SRV) such that only the DB has access to the SRV. In this case, the interface needs only to be documented in the integrator guidance, since untrusted users are unable to gain access.

238 However, consider the case where SRV is now just "somewhere on the network", and now the port that the DB opens up to communicate with the SRV is "exposed" to untrusted users. In this case, while the interface presented by the DB (the TSF) still only needs to be documented in the integrator guidance, additional considerations with respect to vulnerabilities may need to be documented as part of the AVA_VLA activity because of this exposure.

239 In the course of performing its functions, the DB will make system calls down to the OS. This is represented by interface group B2. While these calls are not part of the TSFI, they are an interface that needs to be documented in the integrator guidance.

240 Interface group B3, mentioned previously in connection with interface group A2, is similar to interface group B2 in that these are calls made by the TSF to the IT environment to perform services for the TSF.

241 Having discussed the interfaces in general, the types of TSFI are now discussed in more detail. This discussion categorizes the TSFI into the two categories mentioned previously: TSFI to software directly implementing the SFRs, and TSFI used by administrators.

242 TSFI in the first category are varied in their appearance in a TOE. Most commonly interfaces are thought of as those described in terms of Application Programming Interfaces (APIs), such as kernel calls in a Unix-like operating system. However, interfaces also may be described in terms of menu choices, check boxes, and edit boxes in a GUI; parameter files (the *.INI files and the

registry for Microsoft Windows systems); and network communication protocols at all levels of the protocol stack.

243 TSFI in the second category are more complex. While there are three cases that need to be considered (discussed below), for all cases there is an "additional" requirement that the functions that an administrator uses to perform their duties—as documented in administrative guidance—also are part of the TSFI and must be documented and shown to work correctly. The individual cases are as follows:

- The administrative tool used is also accessible to untrusted users, and runs with some "privilege" itself. In this case the TSFI to be described are similar to those in the first category because the tool itself is privileged.
- The administrative tool uses the privileges of the invoker to perform its tasks. In this case, the interfaces supporting the activities that the administrator is directed to do by the administrative guidance (AGD_ADM, including FMT_* actions) are part of the TSFI. Other interfaces supported by the tool that the administrator is directed not to use (and thus play no role in supporting the TSP), but that are accessible to non-administrators, are not part of the TSFI because there are no privileges associated with their use. Note that this case differs from the previous one in that the tool does not run with privilege, and therefore is not in and of itself interesting from a security point of view. Also note that when FPT_SEP is included in the ST, the executable image of such tools need to be protected so that an untrusted user cannot replace the tool with a "Trojan" tool.
- The administrative tool is only accessible to administrative users. In this case the TSFI are identified in the same manner as the previous case. Unlike the previous case, however, the evaluator ascertains that an untrusted user is unable to invoke the tool when FPT_SEP is included in the ST.

244 It is also important to note that some TOEs will have interfaces that one might consider part of the TSFI, but environmental factors remove them from consideration (an example is the case of interface group B1 discussed earlier). Most of these examples are for TOEs to which untrusted users have restricted access. For example, consider a firewall that untrusted users only have access to via the network interfaces, and further that the network interfaces available only support packet-passing (no remote administration, no firewall-provided services such as telnet). Further suppose that the firewall had a command-line interface that logged-in administrators could use to administer the system, or they could use a GUI-based tool that essentially translated the GUI-based checkboxes, textboxes, etc., into scripts that invoked the command-line utilities. Finally, suppose that the administrators were directed in the administrative guidance to use the GUI-based

tool in administering the firewall. In this case, the command-line interface does not have to be documented because it is inaccessible to untrusted users, and because the administrators are instructed not use it.

245 The term "administrator" above is used in the sense of an entity that has complete trust with respect to all policies implemented by the TSF. There may be entities that are trusted with respect to some policies (e.g., audit) and not to others (e.g., a flow control policy). In these cases, even though the entity may be referred to as an "administrator", they need to be treated as untrusted users with respect to policies to which they have no administrative access. So, in the previous firewall example, if there was an auditor role that was allowed direct log-on to the firewall machine, the command-line interfaces not related to audit is now part of the TSFI, because they are accessible to a user that is not trusted with respect to the policies the interfaces provide access to. The point is that such interfaces need to be addressed in the same manner as previously discussed.

246 Hardware interfaces exist as well. Functions provided by the BIOS of various devices may be visible through a "wrapper" interface such as the IOCTLs in a Unix operating system. If the TOE is or includes a hardware device (e.g., a network interface card), the bus interface signals, and the interface seen at the network port, must be considered "interfaces." Switches that can change the behavior of the hardware are also part of the interface.

247 As indicated above, an interface exists at the TSF boundary if it can be used (by an administrator; untrusted user; or another TOE) to affect the behavior of the TSF. The requirements in this family apply to all types of TSFI, not just APIs.

248 All TSFI are *security relevant*, but some interfaces (or aspects of interfaces) are more critical and require more analysis than other interfaces. If an interface plays a role in enforcing any security policy on the system, then that interface is *security enforcing*. Such policies are not limited to the access control policies, but also refer to any functionality provided by one of the SFRs contained in the ST (with exceptions for FPT_SEP and FPT_RVM as detailed below). Note that it is possible that an interface may have various effects and exceptions, some of which may be security enforcing and some of which may not.

249 FPT_SEP and FPT_RVM are SFRs that require a different type of analysis from other SFRs. These requirements are architecturally related, and their implementation (or lack thereof) is not easily (or efficiently) testable at the TSFI. From a terminology standpoint, although implementation (and the associated analysis) of FPT_SEP and FPT_RVM is critical to the trustworthiness of the system, these two SFRs will not be considered as SFRs that are applicable when

determining the set of security-enforcing TSFIs as defined in the previous paragraph.

250 Interfaces (or parts of an interface) that need only to function correctly in order for the security policies of the system to be preserved are termed *security supporting*. A security supporting interface typically plays a role in supporting the architectural requirements (FPT_SEP or FPT_RVM), meaning that as long as it can be shown that it does not allow the TSF to be compromised or bypassed no further analysis against SFRs is required. In order for an interface to be security supporting it must have *no* security enforcing aspects. In contrast, a security enforcing interface may have security supporting aspects (for example, the ability to set the system clock may be a security enforcing aspect of an interface, but if that same interface is used to display the system date that effect may only be security supporting).

251 A key aspect for the assurance associated with this component is the concept of the evaluator being able to verify that the developer has correctly categorized the security enforcing and security supporting interfaces. The requirements are structured such that the information required for security supporting interfaces is the *minimum* necessary in order for the evaluator to make this determination in an effective manner.

252 For the purposes of the requirements, interfaces are specified (in varying degrees of detail) in terms of their parameters, parameter descriptions, effects, exceptions, and error messages. Additionally, the purpose of each interface, and the way in which the interface is used (both from the point of view of the external stimulus (e.g., the programmer calling the API, the administrator changing a setting in the registry) and the effect on the TSFI that stimulus has) must be specified. This description of method of use must also specify how those administrative interfaces that are unable to be successfully invoked by untrusted users (case “c” mentioned above) are protected.

253 Parameters are explicit inputs to and outputs from an interface that control the behavior of that interface. For examples, parameters are the arguments supplied to an API; the various fields in a packet for a given network protocol; the individual key values in the Windows Registry; the signals across a set of pins on a chip; etc.

254 A parameter description tells what the parameter is in some meaningful way. For instance, the interface “foo(i)” could be described as having “parameter (i) which is an integer”; this is not an acceptable parameter description. A description such as “parameter (i) is an integer that indicates the number of users currently logged in to the system.” is required.

255 Effects of an interface describe what the interface does. The effects that need to be described in an FSP are those that are visible at any external interface, not necessarily limited to the one being specified. For instance, the sole effect of an API call is not just the error code it returns. Also, depending on the parameters of an interface, there may be many different effects (for instance, an API might have the first parameter be a "subcommand", and the following parameters are specific to that subcommand. The IOCTL API in some Unix systems is an example of such an interface).

256 Exceptions refer to the processing associated with "special checks" that may be performed by an interface. An example would be an interface that has a certain set of effects for all users except the Superuser; this would be an exception to the normal effect of the interface. Use of a privilege for some kind of special effect would also be covered in this topic.

257 Documenting the errors associated with the TSF is not as straightforward as it might appear, and deserves some discussion. A general principle is that errors generated by the TSF that are visible to the user should be documented. These errors can be the direct result of invoking a TSFI (an API call that returns an error); an indirect error that is easily tied to a TSFI (setting a parameter in a configuration that is error-checked when read, returning an immediate notification); or an indirect error that is not easily tied to a TSFI (setting a parameter that, in combination with certain system states, generates an error condition that occurs at a later time. An example might be resource exhaustion of a TSF resource due to setting a parameter to too low of a value).

258 Errors can take many forms, depending on the interface being described. For an API, the interface itself may return an error code; set a global error condition, or set a certain parameter with an error code. For a configuration file, an incorrectly configured parameter may cause an error message to be written to a log file. For a hardware PCI card, an error condition may raise a signal on the bus, or trigger an exception condition to the CPU.

259 For the purposes of the requirements, errors are divided into two categories. The first category includes *direct errors,* which are directly related to a TSFI; examples are API calls and parameter-checking for configuration files. For this category of errors, the functional specification must document all of the errors that can be returned as a result of invoking a security-enforcing aspect of the interface such that a reader should be able to associate an interface with the errors it is capable of generating. The second category includes *indirect errors*, which are errors that are not directly tied to the invocation of a TSFI, but which are reported to the user as a result of processing that occurs in the TSF. It should be noted that while the

condition that causes the indirect error can be documented; it is generally much harder to document all the ways in which that condition can occur.[38] Because of the difficulty associated with documenting all of the ways to cause an error and because of the cost of documenting all indirect errors compared to the benefit of having them documented, indirect errors are not required to be documented.

260 The ADV_FSP_(EXP).1.2E element defines a requirement that the evaluator determines that the functional specification is an accurate and complete instantiation of the TOE security functional requirements. This provides a direct correspondence between the TOE security functional requirements and the functional specification, in addition to the pairwise correspondences required by the ADV_RCR family. Although the evaluator may use the evidence provided in ADV_RCR as an input to making this determination, ADV_RCR cannot be the basis for a positive finding in this area. The requirement for completeness is intended to be relative to the level of abstraction of the functional specification.

[38] This may even be impossible, if the error message is for a condition that the programmer does not expect to occur, but is inserted as part of "defensive programming."

## E.3 ADV_HLD_(EXP).1

261 The high-level design of a TOE provides both context for a description of the TSF, and a thorough description of the TSF in terms of major structural units (i.e. subsystems). It relates these units to the functions that they provide. The high-level design requirements are intended to provide assurance that the TOE provides an architecture appropriate to implement the security-enforcing TOE security functional requirements.

262 To provide context for the description of the TSF, the high-level design describes the entire TOE at a high level. From this description the reader should be able to distinguish between the subsystems that are part of the TSF and those that are not. The remainder of the high-level design document then describes the TSF in more detail.

263 The high-level design refines the functional specification into subsystem descriptions. The functional specification provides a description of *what* the TSF does at its interface; the high-level design provides more insight into the TSF by describing *how* the TSF works in order to perform the functions specified at the TSFI. For each subsystem of the TSF, the high-level design identifies the TSFI implemented in the subsystem, describes the purpose of the subsystem and how the implementation of the TSFI (or portions of the TSFI) is designed. The interrelationships of subsystems are also defined in the high-level design. These interrelationships will be represented as data flows, control flows, etc., among the subsystems. It should be noted that this description is at a high level; low-level implementation detail is not necessary at this level of abstraction.

264 The developer is expected to describe the design of the TSF in terms of subsystems. The term “subsystem” is used here to express the idea of decomposing the TSF into a relatively small number of parts. While the developer is not required to actually have “subsystems”, the developer is expected to represent a similar level of decomposition. For example, a design may be similarly decomposed using “layers”, “domains”, or “servers”.

265 A security enforcing subsystem is a subsystem that provides mechanisms for enforcing an element of the TSP, or directly supports a subsystem that is responsible for enforcing the TSP. If a subsystem provides a security-enforcing interface, then the subsystem is security enforcing. If a subsystem does not provide any security enforcing TSFIs, its mechanisms still must preserve the security of the TSF; such subsystems are termed security supporting.

266 As was the case with ADV_FSP_EXP, the set of SFRs that determine the TSP for the purposes of this component do not include FPT_SEP and FPT_RVM. Those two architectural functional requirements require a different type of analysis than that needed for all other SFRs. A security-enforcing subsystem is one that is designed to implement an SFR other than FPT_SEP and FPT_RVM; the design information and justification for the FPT_SEP and FPT_RVM requirements is given as a result of the ADV_ARC_EXP component.

267 The ADV_HLD_EXP component requires that the developer must identify all subsystems of the TSF (not just the security-enforcing ones). In general, the component requires that the security-enforcing aspects of the subsystems be described in more detail than the security-supporting aspects. The descriptions for the security-enforcing aspects should provide the reader with enough information to determine *how* the implementation of the SFRs is designed, while the description for the security-supporting aspects should provide the reader enough assurance to determine that 1) all security-enforcing behavior has been identified and 2) the subsystems or portions of subsystems that are security supporting have been correctly classified.

268 The ADV_HLD_(EXP).1.2E element for this component defines a requirement that the evaluator determine that the high-level design is an accurate and complete instantiation of the user-visible TOE security functional requirements. This provides a direct correspondence between the TOE security functional requirements and the high-level design, in addition to the pair wise correspondences required by the ADV_RCR family. Although the evaluator may use the evidence provided in ADV_RCR as an input to making this determination, ADV_RCR cannot be the basis for a positive finding in this area. The requirement for completeness is intended to be relative to the level of abstraction of the high-level design. Note that for this element FPT_SEP and FPT_RVM are not explicitly analyzed; the analysis for those requirements is done as part of the activity for the ADV_ARC_EXP component.

## E.4 ADV_LLD_(EXP).1

269 The low-level design of a TOE provides a description of the internal workings of the TSF in terms of modules, global data, and their interrelationships. The low-level design is a description of *how* the TSF is implemented to perform its functions, rather than *what* the TSF provides as is specified in the FSP. The low-level design is closely tied to the actual implementation of the TSF, unlike the high-level design, which could be implementation-independent. The primary goal of the low-level design is an aid in understanding the implementation of the TSF, both by reviewing the text of the low-level design as well as a guide when examining the implementation representation (source code).

270 A module is generally a relatively small architectural unit that exhibits properties discussed in ADV_INT_(EXP). A "module" in terms in of the ADV_LLD_EXP requirement refers to the same entity as a "module" for the ADV_INT_EXP requirement.

271 A security-enforcing module is a module that directly implements a security-enforcing TSFI. While this could, for example, include all modules in the call-tree of a security-enforcing module, typically there will be some modules in the call-tree of a security-enforcing module that are not themselves security enforcing. If a module of the TSF is not security enforcing, its implementation still must preserve the security of the TSF; such modules are termed security supporting.

272 A description of a security-enforcing module in the low-level design should be of sufficient detail so that one could create an implementation of the module from the low-level design, and that implementation would

1. be identical to the actual TSF implementation in terms of the interfaces presented and used by the module, and

2. be algorithmically identical to the implementation of the module. For instance, the low-level design may describe a block of processing that is looped over a number of times. The actual implementation may be a *for* loop or a *do* loop, both of which could be used to implement the algorithm. Likewise, a collection of objects could be represented by a linked list or an array; this level of detail is not required to be presented, since both are algorithmically identical. Conversely, if a module's actual implementation performed a bubble sort, it would be inadequate for the low-level design to specify that the module "performed a sort"; it would have to describe the type of sort that was being performed.

273 Security-supporting modules do not need to be described in the same amount of detail, but they should be identified and enough information should be supplied so that 1) the evaluation team can determine that such modules are correctly classified as security supporting (vs. security enforcing), and 2) the evaluation team has the information necessary to complete the analysis required by ADV_INT_(EXP).1.

274 In the low-level design, security-enforcing modules are described in terms of the interfaces they present to other modules; the interfaces they use (call interfaces) from other modules; global data they access; their purpose; and an algorithmic description of how they provide that function. Security supporting modules are described only in terms of the interfaces they present and their purpose.

275 The interfaces presented by a module are those interfaces used by other modules to invoke the functionality provided. Interfaces are described in terms of how their parameters, and any values that are returned from the interface. In addition to a list of parameters, the descriptions of these parameters are also given. If a parameter were expected to take on a set of values (e.g., a "flag" parameter), the complete set of values the parameter could take on that would have an effect on module processing would be specified. Likewise, parameters representing data structures are described such that each field of the data structure is identified and described. Note that different programming languages may have additional "interfaces" that would be non-obvious; an example would be operator/function overloading in C++. This "implicit interface" in the class description would also be described as part of the low-level design. Note that although a module could present only one interface, it is more common that a module presents a small set of related interfaces.

276 By contrast, interfaces used by a module must be identified such that it can be determined the unique interface that is being invoked by the module being described. It must also be clear from the low-level design the algorithmic reason the invoking module is being called. For instance, if Module A is being described, and it uses Module B's bubble sort routine, an inadequate algorithmic description would be "Module A invokes the double_bubble() interface in Module B to perform a bubble sort." An adequate algorithmic description would be "Module A invokes the double_bubble routine with the list of access control entries; double_bubble() will return the entries sorted first on the username, then on the access_allowed field according the following rules..." The low-level design must provide enough detail so that it is clear what effects Module A is expecting from the bubble sort interface. Note that one method of presenting these called interfaces is via a call tree, and then the algorithmic description can be included in the algorithmic description of the called module.

277 If the implementation makes use of global data, the low-level design must describe the global data, and in the algorithmic descriptions of the modules indicate how the specific global data are used by the module. Global data are identified and described much like parameters of an interface.

278 The purpose a module fulfills is a short description indicating what function the module provides. The level of detail provided should be such that the reader could get a general idea of what the module's function is in the architecture, and to determine (for security-supporting modules) that it is not a security-enforcing module.

279 As discussed previously, the algorithmic description of the module should describe in an algorithmic fashion the implementation of the module. This can be done in pseudo-code, through flow charts, or informal text. It discusses how the parameters to the interface, global data, and called functions are used to accomplish the result. It notes changes to global data, system state, and return values produced by the module. It is at the level of detail that an implementation could be derived that would be very similar to the actual implementation of the system. It does not need to describe actual implementation artifacts (*do* loops vs. *for* loops, linked lists vs. arrays) if such artifacts are algorithmically identical.

280 It should be noted that source code does not meet the low-level design requirements. Although the low-level design describes the implementation, it *is not* the implementation. Further, the comments surrounding the source code are not sufficient low-level design if delivered interspersed in the source code. The low-level design must stand on its own, and not depend on source code to provide details that must be provided in the low level design (whether intentionally or unintentionally). However, if the comments were extracted by some automated or manual process to produce the low-level design (independent of the source code statements), they could be found to be acceptable if they met all of the appropriate requirements.

281 The ADV_LLD_(EXP).1.2E element in this component defines a requirement that the evaluator determine that the low-level design is an accurate and complete instantiation of the user-visible TOE security functional requirements. This provides a direct correspondence between the TOE security functional requirements and the low-level design, in addition to the pair-wise correspondences required by the ADV_RCR family. Although the evaluator may use the evidence provided in ADV_RCR as an input to making this determination, ADV_RCR cannot be the basis for a positive finding in this area. The requirement for completeness is intended to be relative to the level of abstraction of the low-level design. Note that for this element, FPT_SEP and FPT_RVM are not

explicitly analyzed; the analysis for those requirements is done as part of the activity for the ADV_ARC_EXP component.

## E.5 ADV_ARC_(EXP).1

282 The architectural design of the TOE is related to the information contained in other decomposition documentation (functional specification, high-level design, low-level design) provided for the TSF, but presents the design in a manner that supports the argument that the TSP cannot be compromised (FPT_SEP) and that it cannot be bypassed (FPT_RVM). The objective of this component is for the developer to provide an architectural design and justification associated with the integrity and non-bypassability properties of the TSF.

283 FPT_SEP and FPT_RVM are distinct from other SFRs because they largely have no directly observable interface at the TSF. Rather, they are properties of the TSF that are achieved through the design of the system, and enforced by the correct implementation of that design. Because of their pervasive nature, the material needed to provide the assurance that these requirements are being achieved is better suited to a presentation separate from the design decomposition of the TSF as embodied in ADV_FSP_EXP, ADV_HLD_EXP, and ADV_LLD_(EXP). This is not to imply that the architectural design called for by this component cannot reference or make use of the design composition material; but it is likely that much of the detail present in the decomposition documentation will not be relevant to the argument being provided for the architectural design document.

284 The architectural design document consists of two types of information. The first is the design information for the entire TSF related to the FPT_SEP and FPT_RVM requirements. This type of information, like the decompositions for ADV_HLD_EXP and ADV_FSP_EXP, describes *how* the TSF is implemented. The description, however, should be focused on providing information sufficient for the reader to determine that the TSF implementation is likely not to be compromised, and that the TSP enforcement mechanisms (that is, those that are implementing SFRs other than FPT_SEP and FPT_RVM) are likely always being invoked.

285 The nature of the FPT_SEP requirement lends itself to a design description much better than FPT_RVM. For FPT_SEP, mechanisms can be identified (e.g., memory management, protected processing modes provided by the hardware, etc.,) and described that implement the domain separation. However, FPT_RVM is concerned with interfaces that bypass the enforcement mechanisms. In most cases this is a consequence of the implementation, where if a programmer is writing an interface that accesses or manipulates an object, it is that programmer's responsibility to use interfaces that are part of the TSP enforcement mechanism for the object and not to try to "go around" those interfaces. However, the developer

is still able to describe architectural elements (e.g., object managers, macros to be invoked for specific functionality) that pertain to the design of the system to achieve the "always invoked" property of the TSF.

286 For FPT_SEP, the design description should cover how user input is handled by privileged-mode routine; what hardware self-protection mechanisms are used and how they work (e.g., memory management hardware, including translation lookaside buffers); how software portions of the TSF use the hardware self-protection mechanisms in providing their functions; and any software protection constructs or coding conventions that contribute to meeting FPT_SEP.

287 For FPT_RVM, the description should cover resources that are protected under the SFRs (usually FDP_* components) and functionality (e.g., audit) that is provided by the TSF. The description should also identify the interfaces that are associated with each of the resources or the functionality; this might make use of the information in the FSP. This description should also describe any design constructs, such as object managers, and their method of use. For instance, if routines are to use a standard macro to produce an audit record, this convention is a part of the design that contributes to the non-bypassability of the audit mechanism. It's important to note that "non-bypassability" in this context is not an attempt to answer the question "could a part of the TSF implementation, if malicious, bypass a TSP mechanism", but rather it's to document how the actual implementation does not bypass the mechanisms implementing the TSP.

288 In addition to the descriptive information indicated in the previous paragraphs, the second type of information an architectural design document must contain is a justification that the FPT_SEP and FPT_RVM requirements are being met. This is distinct from the description, and presents an argument for why the design presented in the description is sufficient.

289 For FPT_SEP, the justification should cover the possible modes by which the TSF could be compromised, and how the mechanisms implemented in response to FPT_SEP counter such compromises. The vulnerability analysis might be referenced in this section.

290 For FPT_RVM, the justification demonstrates that whenever a resource protected by an SFR is accessed, the protection mechanisms of the TSF are invoked (that is, there are no "backdoor" methods of accessing resources that are not identified and analyzed as part of the ADV_FSP_EXP/ADV_HLD_EXP/ADV_LLD_EXP analysis). Similarly, the description demonstrates that a function described by an SFR is always provided where required. For example, if the FCO_NRO family were being used the description should demonstrate that all interfaces either 1) do

not deal with transmitting the information identified in the FCO_NRO component included in the ST, or 2) invoke the mechanism(s) described by the decomposition documentation. The justification for FPT_RVM will likely need to address all of the TSFI in order to make the case that the TSP is non-bypassable.

# F REFINEMENTS

291 This section contains refinements where text was omitted. Omitted text is shown as bold text within parenthesis. The actual text of the functional requirements as presented in Section 5 has been retained.

FAU_ARP.1.1 – **Refinement:** The TSF shall (**take**) [immediately display an alarm message, identifying the potential security violation and make accessible the audit record contents associated with the auditable event(s) that generated the alarm, at the:

- local console,
- remote administrator sessions that exist, and;
- remote administrator sessions that are initiated before the alarm has been acknowledged, and;
- at the option of the Security Administrator, generate an audible alarm, and;
- [assignment: other methods]] upon detection of a potential security violation.

FAU_GEN.1.1-NIAP-0410 – **Refinement:** The TSF shall be able to generate an audit record of the following auditable events:

a) Start-up and shutdown of the audit functions;

b) All auditable events (**for the**) [Table 7] (**level of audit; and**)

c) [selection: [assignment: events at a basic level of audit introduced by the inclusion of additional SFRs determined by the ST Author], [assignment: events commensurate with a basic level of audit introduced by the inclusion of explicit requirements determined by the ST Author], no additional events].

FAU_GEN.1.2-NIAP-0410 – **Refinement:** The TSF shall record within each audit record at least the following information:

a) Date and time of the event, type of event, subject identity (if applicable), and the outcome (success or failure) of the event; and

b) For each audit event type, based on the auditable event definitions of the functional components included in the PP/ST, [(**selection**): [information specified in column three of Error! Reference source not found. below].

FAU_GEN.2.1-NIAP-0410 – **Refinement:** (**For audit events resulting from actions of identified users**,) the TSF shall be able to associate each auditable event with the identity of the user that caused the event.

FAU_SAA.1.2-NIAP-0407 - **Refinement:** The TSF shall **(monitor) enforce the following rules for monitoring audited events**: a) accumulation or combination of [**the following events**:

a) Security Administrator specified number of authentication failures;

b) Security Administrator specified number of Information Flow policy violations by an individual presumed source network identifier (e.g., IP address) within an administrator specified time period;

c) Security Administrator specified number of Information Flow policy violations to an individual destination network identifier within an administrator specified time period;

d) Security Administrator specified number of Information Flow policy violations to an individual destination subject service identifier (e.g., TCP port) within an administrator specified time period;

e) Security Administrator specified Information Flow policy rule, or group of rule violations within an administrator specified time period;

f) Any detected replay of TSF data or security attributes;

g) Any failure of the cryptomodule self-tests (FPT_TST_(EXP).5);

h) Any failure of the other TSF self-tests (FPT_TST_(EXP).4);

i) Security Administrator specified number of encryption failures;

j) Security Administrator specified number of decryption failures] known to indicate a potential security violation;

k) [selection: [assignment: any other rules], "no additional rules"].

FAU_STG.3.1 - **Refinement:** The TSF shall (**take**) [immediately alert the administrators by displaying a message at the local console, and at the remote administrative console when an administrative session exists for each of the defined administrative roles, at the option of the Security Administrator generate an audible alarm, [selection: [assignment: other methods determined by the ST

Author], no other methods] if the audit trail exceeds [a Security Administrator settable percentage of storage capacity].

FAU_STG.1.2-NIAP-0423 – **Refinement:** The TSF shall be able to *prevent* (**unauthorized**) modifications to the audit records in the audit trail.

FCS_CKM.1.1 **Refinement:** The (**TSF**) **cryptomodule** shall generate **symmetric** cryptographic keys (**in accordance with a specified cryptographic key generation algorithm**) [using a FIPS-Approved Random Number Generator] (**and specified cryptographic key sizes**) [for all key sizes] that meet the following: [one of the standards defined in Annex C to FIPS 140-2].

FCS_CKM.4.1 - **Refinement:** The TSF shall destroy cryptographic keys in accordance with a (**cryptographic key destruction method**) that meets the following:

a) [The Key Zeroization Requirements in FIPS PUB 140-2 Key Management Security Levels 3;

b) Zeroization of all private cryptographic keys, plaintext cryptographic keys and all other critical cryptographic security parameters shall be immediate and complete; and

c) The zeroization shall be executed by overwriting the key/critical cryptographic security parameter storage area three or more times with an alternating pattern.

d) The TSF shall overwrite each intermediate storage area for private cryptographic keys, plaintext cryptographic keys, and all other critical security parameters three or more times with an alternating pattern upon the transfer of the key/CSPs to another location.]

FCS_COP.1.1(2) **Refinement**: The TSF shall perform **cryptographic signature services** in accordance with the **NIST-approved digital signature** algorithm **[selection*:***

a) Digital Signature Algorithm (DSA) with a key size (modulus) of 2048 bits or greater,

b) RSA Digital Signature Algorithm (rDSA with odd e) with a key size (modulus) of 2048 bits or greater, or

c) Elliptic Curve Digital Signature Algorithm (ECDSA) with a key size of 256 bits or greater]

292 *Application Note: For elliptic curve-based schemes the key size refers to the log2 of the order of the base point. As the preferred approach for cryptographic signature, elliptic curves will be required within a TBD time frame after all the necessary standards and other supporting information are fully established.*

that meets the following:

a) **Case: Digital Signature Algorithm**
**FIPS PUB 186-2, Digital Signature Standard, for signature creation and verification processing; and ANSI Standard X9.42-2001, Public Key Cryptography for the Financial Services Industry: Agreement of Symmetric Keys Using Discrete Logarithm Cryptography for generation of the domain parameters;**

b) **Case: RSA Digital Signature Algorithm (with odd e)**
**ANSI X 9.31-1998 (May 1998), Digital Signatures Using Reversible Public Key Cryptography For The Financial Services Industry (rDSA);**

c) **Case: Elliptic Curve Digital Signature Algorithm**
**ANSI X9.62-1-xxxx (10 Oct 1999), Public Key Cryptography for the Financial Services Industry: Elliptic Curve Digital Signature Algorithm (ECDSA).**

FCS_COP.1.1(4) **Refinement**: The TSF shall perform **cryptographic key agreement services** in accordance with a **NIST-approved implementation of a key agreement** algorithm ***[selection****:*

*a) Finite Field-based key agreement algorithm and cryptographic key sizes(modulus) of 2048 bits or greater,*

*b) Elliptic Curve-based key agreement algorithm and cryptographic key size of 256 bits or greater]*

293 *Application Note: For elliptic curve-based schemes the key size refers to the log2 of the order of the base point. As the preferred approach for key exchange, elliptic curves will be required within a TBD time frame after all the necessary standards and other supporting information are fully established.*

that meets the following:

a) **Case: Finite field-based key agreement schemes**
**ANSI X9.42-2001, Public Key Cryptography for the Financial Services Industry: Agreement of Symmetric Keys Using Discrete Logarithm Cryptography;**

294 *Application Note: For example, "Classic" Diffie-Hellman-based schemes*

b) **Case: Elliptic curve-based key agreement schemes**
**ANSI X9.63-200x (1 Oct 2000), Public Key Cryptography for the Financial Services Industry: Key Agreement and Key Transport using Elliptic Curve Cryptography.**

295 *Application Note: Some authentication mechanism on the keying material is recommended. In addition, repeated generation of the same shared secrets should be avoided. As an example, the MQV schemes described in the above standards address these issues.*

FDP_IFF.1.2(1) - **Refinement**: The TSF shall permit an information flow between a **source (controlled)** subject and a **destination subject (controlled information)** via a controlled operation if the following rules hold:

- [the presumed identity of the source subject is in the set of source subject identifiers;
- the identity of the destination subject is in the set of source destination identifiers;
- the information security attributes match the attributes in an information flow policy rule (contained in the information flow policy ruleset defined by the Security Administrator) according to the following algorithm [assignment: algorithm used by the TOE to match information security attributes to information flow policy rules]; and
- the selected information flow policy rule specifies that the information flow is to be permitted].

FDP_IFF.1.2(2) - **Refinement**: The TSF shall permit an information flow between a **source** subject and a **destination subject** via a controlled operation if the following rules hold:

- [the source subject has successfully authenticated to the TOE;
- the identity of the destination subject is in the set of destination identifiers;
- the information security attributes match the attributes in a information flow policy rule (contained in the information flow policy ruleset defined by the administrator) according to the following algorithm [assignment: algorithm used by the TOE to match information security attributes to information flow policy rules]; and

- the selected information flow policy rule specifies that the information flow is to be permitted].

FIA_AFL.1.2-NIAP-0425 – **Refinement**: When the defined number of unsuccessful authentication attempts has been met (**or surpassed**), the TSF shall [at the option of the Security Administrator prevent the remote administrators or authorized IT entity from performing activities that require authentication until an action is taken by the Security Administrator, or until a Security Administrator defined time period has elapsed].

FIA_ATD.1.1(1) **Refinement:** The TSF shall maintain the following list of security attributes belonging to (**individual users) authorized user**:

a) [user identifier(s):

role;

[selection: [assignment: Any security attributes related to a user identifier (e.g., certificate associated with the userid)], none]; and

b) [selection: [assignment: other user security attributes], none]].

FIA_ATD.1.1(2) **Refinement:** The TSF shall maintain the following list of security attributes belonging to **(individual users) authorized subjects**:

a) [subject identity;

b) [assignment: any other security attributes].

FMT_MSA.3.1-NIAP-0409(1) – **Refinement**: The TSF shall enforce the [UNAUTHENTICATED INFORMATION FLOW SFP, AUTHENTICATED INFORMATION FLOW SFP] to provide restrictive default values for (**security attributes) the information flow policy ruleset** that (**are**) is used to enforce the SFP.

FMT_MSA.3.1-NIAP-0409(2) – **Refinement**: The TSF shall enforce the [UNAUTHENTICATED TOE SERVICES SFP] to provide restrictive default values for **(security attributes) (that are used to enforce the SFP) the set of TOE services available to unauthenticated users**.

FMT_REV.1.2 - **Refinement**: The TSF shall **immediately** enforce the (**rules**):

- [revocation of a user's role (Security Administrator, Cryptographic Administrator, Audit Administrator);
- changes to the information flow policy ruleset when applied;
- disabling of a service available to unauthenticated users; and
- [selection: [assignment: other rules as determined by the ST Author], none]].

FPT_SEP.2.3 - **Refinement**: The TSF shall maintain the part of the TSF related to [cryptography] in (**security domain(s)) an address space** for **(their) its** own execution that protects **(them) it** from interference and tampering by the remainder of the TSF and by subjects untrusted with respect to **(those SFPs) the cryptographic functionality.**

FTA_SSL.1.2 - **Refinement:** The TSF shall require the **(following events to occur) user to re-authenticate** prior to unlocking the session(: **[assignment: events to occur]**).

FTA_SSL.2.2 - **Refinement**: The TSF shall require the (**following events to occur) user to re-authenticate** prior to unlocking the session(: **[assignment: events to occur]**).

FTA_TSE.1.1 - **Refinement**: The TSF shall be able to deny **(session)** establishment **of an authorized user session** based on [location, time, and day].

FTP_ITC.1.1(1) - **Refinement**: The TSF shall **(provide) use encryption to** provide a **trusted** communication channel between itself and **(a remote trusted IT product) authorized IT entities** that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from (**modification or**) disclosure.

FTP_ITC.1.1(2) - **Refinement**: The TSF shall **(provide) use a cryptographic signature** to provide a **trusted** communication channel between itself and **(a remote trusted IT product) authorized IT entities** that is logically distinct from other communication channels and provides assured identification of its end points and **(protection of the channel data from modification or disclosure) detection of the modification of data**.

FTP_TRP.1.1(1) - **Refinement**: The TSF shall provide **(a) an encrypted** communication path between itself and remote **administrators and authenticated** users that is logically distinct from other communication paths and provides assured identification of its end points and protection of the communicated data from (**modification or**) disclosure.

FTP_TRP.1.1(2) - **Refinement**: The TSF shall **use a cryptographic signature to** provide a communication path between itself and remote **administrators and authenticated** users that is logically distinct from other communication paths and provides assured identification of its end points and (**protection**) **detection (of the communicated data from) modification (or disclosure) of data**.

FTP_ITC.1.1(1) - **Refinement**: The **(TSF) IT Environment** shall provide a **trusted** communication channel between itself and the **(a remote trusted IT product) TSF** that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from (**modification or**) disclosure.

FTP_ITC.1.1(2) - **Refinement:** The **(TSF) IT Environment** shall provide **(a) an encrypted** communication channel between itself and **(a remote trusted IT product) the TSF** that is logically distinct from other communication channels and provides assured identification of its end points and **(protection of the channel data from modification or disclosure) detection of the modification of data.**

FTP_TRP.1.1(1) - **Refinement**: The **(TSF) IT Environment** shall provide **(a) an encrypted** communication path between itself and ([**selection: remote, local**]) **the TSF (users)** that is logically distinct from other communication paths and provides assured identification of its end points and protection of the communicated data from modification or disclosure.

FTP_TRP.1.2(1) - The **(TSF) IT Environment** shall permit remote users **of the TSF** to initiate communication **to the TSF** via the trusted path.

FTP_TRP.1.3(1) – **Refinement**: The **(TSF) IT Environment** shall **(require) initiate** the use of the trusted path for ***(initial)*** *user authentication,* ***all remote administration actions,*** *[selection: [assignment: other services for which trusted path is required], none]*.

FTP_TRP.1.1(2) - **Refinement**: The **(TSF) IT Environment** shall provide **(a)** an **encrypted** communication path between itself and ([**selection: remote, local**]) **(users) the TSF** that is logically distinct from other communication paths and provides assured identification of its end points **(protection of the communicated data from modification or disclosure) and detection of the modification of data.**

FTP_TRP.1.2(2) - The **(TSF) IT Environment** shall permit *remote users* ***of the TSF*** to initiate communication **to the TSF** via the trusted path.

FTP_TRP.1.3(2) – **Refinement**: The **(TSF) IT Environment** shall **(require) initiate** the use of the trusted path for ***(initial)*** *user authentication,* ***all remote administration actions****, [selection: [assignment: other services for which trusted path is required], none]*.

# G STATISTICAL RANDOM NUMBER GENERATOR TESTS

A cryptographic module employing random number generators (RNGs) shall perform the following statistical tests for randomness. A single bit stream of 20,000 consecutive bits of output from each RNG shall be subjected to the following four tests: monobit test, poker test, runs test, and long runs test. (These four tests are simply those that formerly existed as the statistical RNG tests in Federal Information Processing Standard 140-2. However, for purposes of meeting this protection profile, these tests must be performed at the frequency specified earlier in this protection profile.)

The Monobit Test:

1. Count the number of ones in the 20,000 bit stream. Denote this quantity by X.
2. The test is passed if $9,725 < X < 10,275$.

The Poker Test:

1. Divide the 20,000 bit stream into 5,000 contiguous 4 bit segments. Count and store the number of occurrences of the 16 possible 4 bit values. Denote f(i) as the number of each 4 bit value i, where $0 < i < 15$.
2. Evaluate the following:

$$X = (16 / 5000) * (\sum_{i=0} [f(i)]2 ) – 5000$$

3. The test is passed if $2.16 < X < 46.17$.

The Runs Test:

1. A run is defined as a maximal sequence of consecutive bits of either all ones or all zeros that is part of the 20,000 bit sample stream. The incidences of runs (for both consecutive zeros and consecutive ones) of all lengths (> 1) in the sample stream should be counted and stored.
2. The test is passed if the runs that occur (of lengths 1 through 6) are each within the corresponding interval specified in the table below. This must hold for both the zeros and ones (i.e., all 12 counts must lie in the specified interval). For the purposes of this test, runs of greater than 6 are considered to be of length 6.

Table C.1 - Required Intervals for Length of Runs Test

| Length of Run | Required Interval |
|---|---|
| 1 | 2343 – 2657 |
| 2 | 1135 – 1365 |
| 3 | 542 - 708 |
| 4 | 251 - 373 |

| | |
|---|---|
| 5 | 111 - 201 |
| 6 and greater | 111 - 201 |

The Long Runs Test:
1. A long run is defined to be a run of length 26 or more (of either zeros or ones).
2. On the sample of 20,000 bits, the test is passed if there are no long runs.

# H RANDOMIZER QUALIFICATION TESTING REQUIREMENTS

This test utilizes the NIST battery of statistical tests as described in "A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications", NIST Special Publication 800-22. This document and corresponding software code are available for downloading at the following Internet sites: http://csrc.ncsl.nist.gov/rng or http://csrc.ncsl.nist.gov/CryptoToolkit/tkrng .
297 The Randomizer Qualification Statistical Test Suite consists of the following statistical tests:

1. Frequency (Monobit) Test
2. Frequency Test within a Block
3. Cumulative Suns (Cusum) Test
4. Runs Test
5. Longest Run of ones in a Block
6. Binary Matrix Rank Test
7. Discrete Fourier Transform (Spectral) Test
8. Maurer's Universal Statistical Test
9. Approximate Entropy Test
10. Serial Test

Randomizer Qualification Test Process
Power up the randomizer and collect a sample of 100,000 bits of data every 5 minutes until 10 samples have been collected. Concatenate the 10 samples to form a single sample of length 1,000,000 bits. Apply the above statistical tests using the following input parameters:
Sequence Length: 100,000
Number of Sequences: 10
Block Frequency Test Block Length: 100
Universal Test Block Length: 6
Universal Test Number of Initialization Steps: 640
Approximate Entropy Block Length: 10
Serial Test Block Length: 10

Each statistical test will produce a series of 10 P-Values. The Cusum and Serial test consist of two tests each and produces two series of 10 P-Values each. Thus the statistical test suite will produce twelve series of 10 P-Values each. The collected sample of data passes the statistical test suite if for each of the twelve series of P-Values at least 9 of the 10 P-Values are greater than 0.01. The NIST software generates a file, finalAnalysisReport, which summarizes the results of the tests. The data passes the

statistical test suite if all of the twelve values listed in the proportions column are greater than or equal to 0.9.

The above test procedure is to be repeated 3 times. The randomizer passes the randomizer qualification test if the statistical test suite is passes on at least 2 of the 3 attempts.